Fast Models

Version 11.13

Reference Manual

arm

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
100964_1113_00_en

Fast Models

Reference Manual

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.

Release Information

Document History

Issue	Date	Confidentiality	Change
A	31 May 2014	Non-Confidential	New document for Fast Models v9.0, from DUI0423Q for v8.3.
B	30 November 2014	Non-Confidential	Update for v9.1.
C	28 February 2015	Non-Confidential	Update for v9.2.
D	31 May 2015	Non-Confidential	Update for v9.3.
E	31 August 2015	Non-Confidential	Update for v9.4.
F	30 November 2015	Non-Confidential	Update for v9.5.
G	29 February 2016	Non-Confidential	Update for v9.6.
H	31 May 2016	Non-Confidential	Update for v10.0.
I	31 August 2016	Non-Confidential	Update for v10.1.
J	11 November 2016	Non-Confidential	Update for v10.2.
K	17 February 2017	Non-Confidential	Update for v10.3.
1100-00	31 May 2017	Non-Confidential	Update for v11.0. Document numbering scheme has changed.
1101-00	31 August 2017	Non-Confidential	Update for v11.1.
1102-00	17 November 2017	Non-Confidential	Update for v11.2.
1103-00	23 February 2018	Non-Confidential	Update for v11.3.
1104-00	22 June 2018	Non-Confidential	Update for v11.4.
1104-01	17 August 2018	Non-Confidential	Update for v11.4.2.
1105-00	23 November 2018	Non-Confidential	Update for v11.5.
1106-00	27 February 2019	Non-Confidential	Update for v11.6.
1106-01	25 March 2019	Non-Confidential	Update for v11.6.1.
1107-00	17 May 2019	Non-Confidential	Update for v11.7.
1108-00	05 September 2019	Non-Confidential	Update for v11.8.
1109-00	28 November 2019	Non-Confidential	Update for v11.9.
1110-00	12 March 2020	Non-Confidential	Update for v11.10.
1111-00	09 June 2020	Non-Confidential	Update for v11.11.
1112-00	22 September 2020	Non-Confidential	Update for v11.12.
1113-00	09 December 2020	Non-Confidential	Update for v11.13.

Non-Confidential Proprietary Notice

This document is protected by copyright and other related rights and the practice or implementation of the information contained in this document may be protected by one or more patents or pending patent applications. No part of this document may be reproduced in any form by any means without the express prior written permission of Arm. **No license, express or implied, by estoppel or otherwise to any intellectual property rights is granted by this document unless specifically stated.**

Your access to the information in this document is conditional upon your acceptance that you will not use or permit others to use the information for the purposes of determining whether implementations infringe any third party patents.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

THIS DOCUMENT IS PROVIDED “AS IS”. ARM PROVIDES NO REPRESENTATIONS AND NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY, SATISFACTORY QUALITY, NON-INFRINGEMENT OR FITNESS FOR A PARTICULAR PURPOSE WITH RESPECT TO THE DOCUMENT. For the avoidance of doubt, Arm makes no representation with respect to, and has undertaken no analysis to identify or understand the scope and content of, third party patents, copyrights, trade secrets, or other rights.

This document may include technical inaccuracies or typographical errors.

TO THE EXTENT NOT PROHIBITED BY LAW, IN NO EVENT WILL ARM BE LIABLE FOR ANY DAMAGES, INCLUDING WITHOUT LIMITATION ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, PUNITIVE, OR CONSEQUENTIAL DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF ANY USE OF THIS DOCUMENT, EVEN IF ARM HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

This document consists solely of commercial items. You shall be responsible for ensuring that any use, duplication or disclosure of this document complies fully with any relevant export laws and regulations to assure that this document or any portion thereof is not exported, directly or indirectly, in violation of such export laws. Use of the word “partner” in reference to Arm’s customers is not intended to create or refer to any partnership relationship with any other company. Arm may make changes to this document at any time and without notice.

If any of the provisions contained in these terms conflict with any of the provisions of any click through or signed written agreement covering this document with Arm, then the click through or signed written agreement prevails over and supersedes the conflicting provisions of these terms. This document may be translated into other languages for convenience, and you agree that if there is any conflict between the English version of this document and any translation, the terms of the English version of the Agreement shall prevail.

The Arm corporate logo and words marked with ® or ™ are registered trademarks or trademarks of Arm Limited (or its subsidiaries) in the US and/or elsewhere. All rights reserved. Other brands and names mentioned in this document may be the trademarks of their respective owners. Please follow Arm’s trademark usage guidelines at *http://www.arm.com/company/policies/trademarks*.

Copyright © 2014–2020 Arm Limited (or its affiliates). All rights reserved.

Arm Limited. Company 02557590 registered in England.

110 Fulbourn Road, Cambridge, England CB1 9NJ.

(LES-PRE-20349)

Confidentiality Status

This document is Non-Confidential. The right to use, copy and disclose this document may be subject to license restrictions in accordance with the terms of the agreement entered into by Arm and the party that Arm delivered this document to.

Unrestricted Access is an Arm internal classification.

Product Status

The information in this document is Final, that is for a developed product.

Web Address

developer.arm.com

Progressive terminology commitment

Arm values inclusive communities. Arm recognizes that we and our industry have used terms that can be offensive. Arm strives to lead the industry and create change.

This document includes terms that can be offensive. We will replace these terms in a future issue of this document.

If you find offensive terms in this document, please contact *terms@arm.com*.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Contents
Fast Models Reference Manual

Preface

About this book 10

Chapter 1 Introduction

1.1 About the models 1-13
1.2 Model capabilities 1-14
1.3 Fast Models accuracy 1-15
1.4 Processor implementation 1-19
1.5 Fast Models CADI implementation 1-25
1.6 CADI interactions with processor behavior 1-27
1.7 CADI sync watchpoints 1-28
1.8 Non-CADI sync watchpoints 1-29
1.9 SCADI 1-32
1.10 Using the TelnetTerminal 1-36
1.11 Network set up 1-38
1.12 Model Trace Interface 1-46
1.13 Using parameters to set port values 1-48
1.14 About PVBus system components 1-49
1.15 PVBus C++ transaction and Tx_Result classes 1-50
1.16 Visualisation Library 1-53

Chapter 2 Protocols

2.1 AMBA-PV protocols 2-59
2.2 Clocking protocols 2-65

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

2.3 Debug interface protocols 2-67
2.4 Peripheral protocols 2-71
2.5 Power management protocols 2-80
2.6 Processor protocols 2-81
2.7 Signaling protocols 2-85

Chapter 3 Fast Models components

3.1 Bridge components 3-88
3.2 Bus components 3-102
3.3 Clocking components 3-113
3.4 Core components 3-117
3.5 Media components 3-601
3.6 Other components 3-616
3.7 Peripheral components 3-618
3.8 Signals components 3-644
3.9 SystemIP components 3-647

Chapter 4 Trace components

4.1 AHCI_SATA - trace 4-864
4.2 AMBAPVSignal2SGSignalx8 - trace 4-869
4.3 ARM_AEM-A_MP - trace 4-870
4.4 ARM_AEMv8-A_MP - trace 4-908
4.5 ARM_AEMv8M - trace 4-946
4.6 ARM_Cortex-A15 - trace 4-979
4.7 ARM_Cortex-A17 - trace 4-1015
4.8 ARM_Cortex-A32 - trace 4-1051
4.9 ARM_Cortex-A35 - trace 4-1089
4.10 ARM_Cortex-A5 - trace 4-1127
4.11 ARM_Cortex-A53 - trace 4-1152
4.12 ARM_Cortex-A55 - trace 4-1190
4.13 ARM_Cortex-A57 - trace 4-1228
4.14 ARM_Cortex-A5MP - trace 4-1266
4.15 ARM_Cortex-A65 - trace 4-1291
4.16 ARM_Cortex-A65AE - trace 4-1329
4.17 ARM_Cortex-A7 - trace 4-1367
4.18 ARM_Cortex-A72 - trace 4-1403
4.19 ARM_Cortex-A73 - trace 4-1441
4.20 ARM_Cortex-A75 - trace 4-1479
4.21 ARM_Cortex-A76 - trace 4-1517
4.22 ARM_Cortex-A76AE - trace 4-1556
4.23 ARM_Cortex-A77 - trace 4-1595
4.24 ARM_Cortex-A78 - trace 4-1634
4.25 ARM_Cortex-A78C - trace 4-1673
4.26 ARM_Cortex-A8 - trace 4-1712
4.27 ARM_Cortex-A9MP - trace 4-1736
4.28 ARM_Cortex-A9UP - trace 4-1772
4.29 ARM_Cortex-M0 - trace 4-1808
4.30 ARM_Cortex-M0+ - trace 4-1839
4.31 ARM_Cortex-M23 - trace 4-1870
4.32 ARM_Cortex-M3 - trace 4-1901

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.33 ARM_Cortex-M33 - trace 4-1921
4.34 ARM_Cortex-M35P - trace 4-1954
4.35 ARM_Cortex-M4 - trace 4-1987
4.36 ARM_Cortex-M55 - trace 4-2009
4.37 ARM_Cortex-M7 - trace 4-2047
4.38 ARM_Cortex-R4 - trace 4-2080
4.39 ARM_Cortex-R5 - trace 4-2103
4.40 ARM_Cortex-R7 - trace 4-2126
4.41 ARM_Cortex-R8 - trace 4-2161
4.42 ARM_Cortex-X1 - trace 4-2196
4.43 ARM_CortexR52 - trace 4-2235
4.44 ARM_Neoverse-E1 - trace 4-2273
4.45 ARM_Neoverse-N1 - trace 4-2311
4.46 ARM_SC000 - trace 4-2350
4.47 ARM_SC300 - trace 4-2381
4.48 ARMv8Cluster - trace 4-2401
4.49 AsyncCacheFlushUnit - trace 4-2402
4.50 Base_PowerController - trace 4-2403
4.51 CCI400 - trace 4-2405
4.52 CCI500 - trace 4-2407
4.53 CCI550 - trace 4-2412
4.54 CCNCache - trace 4-2417
4.55 CCNRegisterSet - trace 4-2422
4.56 CMN600 - trace 4-2424
4.57 CMN600CML - trace 4-2428
4.58 CMN600Cache - trace 4-2429
4.59 CMN650 - trace 4-2434
4.60 ClockDivider - trace 4-2438
4.61 D71 - trace 4-2439
4.62 DMC-500 - trace 4-2448
4.63 DMC-520 - trace 4-2450
4.64 DMC-620 - trace 4-2452
4.65 DP500 - trace 4-2455
4.66 DP550 - trace 4-2457
4.67 DP650 - trace 4-2460
4.68 DSU - trace 4-2463
4.69 DualClusterSystemConfigurationBlock - trace 4-2464
4.70 GIC_400 - trace 4-2466
4.71 GICv2 - trace 4-2470
4.72 GICv3CPUInterface - trace 4-2474
4.73 GICv3CPUInterfaceDecoder - trace 4-2489
4.74 GICv3CommsLogger - trace 4-2490
4.75 GICv3CommsPVBus - trace 4-2494
4.76 GICv3Distributor - trace 4-2496
4.77 GICv3IRI - trace 4-2504
4.78 GICv3InterruptTranslationService - trace 4-2505
4.79 GICv3ProtocolChecker - trace 4-2526
4.80 GICv3Redistributor - trace 4-2527
4.81 GICv3RedistributorInternal - trace 4-2541
4.82 MMU_400_BASE - trace 4-2547

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.83 MMU_500_BASE - trace 4-2576
4.84 MMU_600 - trace 4-2605
4.85 MMU_700 - trace 4-2664
4.86 MSIRewriter - trace 4-2723
4.87 Mali_G51 - trace 4-2724
4.88 Mali_G71 - trace 4-2726
4.89 Mali_G72 - trace 4-2728
4.90 Mali_G76 - trace 4-2730
4.91 MemoryMappedCounterModule - trace 4-2732
4.92 MessageHandlingUnitV2 - trace 4-2733
4.93 NonVolatileCounter - trace 4-2736
4.94 PL011_Uart - trace 4-2737
4.95 PL11x_CLCD - trace 4-2738
4.96 PL310_L2CC - trace 4-2739
4.97 PL330_DMAC - trace 4-2742
4.98 PPUv0 - trace 4-2747
4.99 PPUv1 - trace 4-2748
4.100 PVBus2AMBAPVACE - trace 4-2750
4.101 PVBusBridge - trace 4-2751
4.102 PVBusExclusiveMonitor - trace 4-2752
4.103 PVBusGICv3Comms - trace 4-2754
4.104 PVBusLogger - trace 4-2756
4.105 PVBusMapper - trace 4-2758
4.106 PVBusMaster - trace 4-2759
4.107 PVBusSlave - trace 4-2760
4.108 PVCache - trace 4-2761
4.109 PVWriteBuffer - trace 4-2767
4.110 RAMDevice - trace 4-2769
4.111 SMMUv3AEM - trace 4-2770
4.112 SMMUv3TestEngine - trace 4-2827
4.113 SignalLogger - trace 4-2833
4.114 TLB - trace 4-2834
4.115 V550 - trace 4-2838
4.116 V61 - trace 4-2852
4.117 Value64Logger - trace 4-2866
4.118 ValueLogger - trace 4-2867
4.119 VirtioBlockDevice - trace 4-2868
4.120 VirtioP9Device - trace 4-2871
4.121 VirtioPCIBlockDevice - trace 4-2872
4.122 VirtualEthernetCrossover - trace 4-2875
4.123 atc - trace 4-2876
4.124 v7_VGIC - trace 4-2886

Chapter 5 Plug-ins for Fast Models

5.1 Loading a plug-in 5-2891
5.2 ArchMsgTrace 5-2893
5.3 BranchPrediction 5-2894
5.4 CADIIPCRemoteConnection 5-2898
5.5 CDE 5-2899
5.6 Crypto 5-2909

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

5.7 Fastline 5-2911
5.8 GDBRemoteConnection 5-2919
5.9 GenericCounter 5-2921
5.10 GenericTrace 5-2922
5.11 ListTraceSources 5-2924
5.12 PipelineModel 5-2925
5.13 RunTimeParameterTest 5-2930
5.14 ScalableVectorExtension 5-2931
5.15 Sidechannel 5-2934
5.16 TarmacText 5-2935
5.17 TarmacTrace 5-2936

Chapter 6 Fast Models examples
6.1 CADI examples 6-2955
6.2 LISA examples 6-2956
6.3 LISAPlus examples 6-2957
6.4 MTI examples 6-2958
6.5 Python examples 6-2959
6.6 SystemCExport examples 6-2960

Chapter 7 Base Platform
7.1 Base - about 7-2962
7.2 Base Platform RevC 7-2963
7.3 Base - memory 7-2967
7.4 Base - interrupt assignments 7-2971
7.5 Base - clocks 7-2974
7.6 Base - parameters 7-2975
7.7 Base - components 7-2976
7.8 Base - differences between the AEMv8-A FVP and core FVPs 7-2984
7.9 Base - VE compatibility 7-2985
7.10 Base - unsupported VE features 7-2987

Chapter 8 Microcontroller Prototyping System 2
8.1 MPS2 - about 8-2989
8.2 MPS2 platform types 8-2990
8.3 MPS2 - memory maps 8-2991
8.4 MPS2 - interrupt assignments 8-2997
8.5 MPS2 - differences between models and hardware 8-2998

Chapter 9 Versatile Express Model
9.1 VE - about 9-3000
9.2 VE memory map for Cortex®-A series 9-3002
9.3 VE memory map for Cortex®-R series 9-3005
9.4 VE - interrupt assignments for Cortex®-A series 9-3006
9.5 VE - interrupt assignments for Cortex®-R series 9-3007
9.6 VE parameters 9-3009
9.7 VE - components 9-3011
9.8 Differences between the VE hardware and the system models 9-3017

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Preface

This preface introduces the *Fast Models Reference Manual*.

It contains the following:

- *About this book* on page 10.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

About this book

This document provides a reference for signaling, clock, bus, generic peripheral, and processor components included in Fast Models.

Using this book

This book is organized into the following chapters:

Chapter 1 Introduction

This chapter introduces the document.

Chapter 2 Protocols

Components communicate through connected ports. Ports have protocols that define the function calls for different connections.

Chapter 3 Fast Models components

This chapter describes all model components in Fast Models, organized by component type.

Chapter 4 Trace components

This chapter describes the trace components that are provided by Fast Models components.

Chapter 5 Plug-ins for Fast Models

This chapter describes the plug-ins that are available for Fast Models.

Chapter 6 Fast Models examples

The following top-level example directories are installed in `$PVLIB_HOME/examples`.

Chapter 7 Base Platform

This chapter describes the Base Platform system model.

Chapter 8 Microcontroller Prototyping System 2

This chapter describes the MPS2 system model.

Chapter 9 Versatile Express Model

This chapter describes the components of Fast Models that are specific to the Versatile Express (VE) model of the hardware platform.

Glossary

The Arm® Glossary is a list of terms used in Arm documentation, together with definitions for those terms. The Arm Glossary does not contain terms that are industry standard unless the Arm meaning differs from the generally accepted meaning.

See the *Arm® Glossary* for more information.

Typographic conventions

italic
: Introduces special terminology, denotes cross-references, and citations.

bold
: Highlights interface elements, such as menu names. Denotes signal names. Also used for terms in descriptive lists, where appropriate.

`monospace`
: Denotes text that you can enter at the keyboard, such as commands, file and program names, and source code.

`mono`space
: Denotes a permitted abbreviation for a command or option. You can enter the underlined text instead of the full command or option name.

`monospace italic`
: Denotes arguments to monospace text where the argument is to be replaced by a specific value.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

`monospace bold`
Denotes language keywords when used outside example code.

`<and>`
Encloses replaceable terms for assembler syntax where they appear in code or code fragments. For example:

```
MRC p15, 0, <Rd>, <CRn>, <CRm>, <Opcode_2>
```

SMALL CAPITALS
Used in body text for a few terms that have specific technical meanings, that are defined in the *Arm® Glossary*. For example, IMPLEMENTATION DEFINED, IMPLEMENTATION SPECIFIC, UNKNOWN, and UNPREDICTABLE.

Feedback

Feedback on this product

If you have any comments or suggestions about this product, contact your supplier and give:

- The product name.
- The product revision or version.
- An explanation with as much information as you can provide. Include symptoms and diagnostic procedures if appropriate.

Feedback on content

If you have comments on content then send an e-mail to *errata@arm.com*. Give:

- The title *Fast Models Reference Manual*.
- The number 100964_1113_00_en.
- If applicable, the page number(s) to which your comments refer.
- A concise explanation of your comments.

Arm also welcomes general suggestions for additions and improvements.

Note

Arm tests the PDF only in Adobe Acrobat and Acrobat Reader, and cannot guarantee the quality of the represented document when used with any other PDF reader.

Other information

- *Arm® Developer*.
- *Arm® Documentation*.
- *Technical Support*.
- *Arm® Glossary*.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Chapter 1
Introduction

This chapter introduces the document.

It contains the following sections:

- *1.1 About the models* on page 1-13.
- *1.2 Model capabilities* on page 1-14.
- *1.3 Fast Models accuracy* on page 1-15.
- *1.4 Processor implementation* on page 1-19.
- *1.5 Fast Models CADI implementation* on page 1-25.
- *1.6 CADI interactions with processor behavior* on page 1-27.
- *1.7 CADI sync watchpoints* on page 1-28.
- *1.8 Non-CADI sync watchpoints* on page 1-29.
- *1.9 SCADI* on page 1-32.
- *1.10 Using the TelnetTerminal* on page 1-36.
- *1.11 Network set up* on page 1-38.
- *1.12 Model Trace Interface* on page 1-46.
- *1.13 Using parameters to set port values* on page 1-48.
- *1.14 About PVBus system components* on page 1-49.
- *1.15 PVBus C++ transaction and Tx_Result classes* on page 1-50.
- *1.16 Visualisation Library* on page 1-53.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

1.1 About the models

Programmer's View (PV) models of processors and devices work at a level where functional behavior is equivalent to what a programmer would see using the hardware.

They sacrifice timing accuracy to achieve fast simulation execution speeds: you can use the PV models for confirming software functionality, but you must not rely on the accuracy of cycle counts, low-level component interactions, or other hardware-specific behavior.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

1.2 Model capabilities

Fast Models attempt to accurately model the hardware, but compromises exist between speed of execution, accuracy and other criteria. A processor model might not match the hardware under certain conditions.

Fast Models can:

- Accurately model instructions.
- Correctly execute architecturally-correct code.
- Model some unpredictable behavior.

Fast Models cannot:

- Validate the hardware.
- Model all unpredictable behavior.
- Model cycle counting.
- Model timing sensitive behavior.
- Model SMP instruction scheduling.
- Measure software performance.
- Model bus traffic.

Related concepts

1.3.1 How accurate are Fast Models? on page 1-15

Related references

1.3.6 Caches in Fast Models on page 1-17

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

1.3 Fast Models accuracy

This section describes the aspects of Fast Models accuracy, and specific areas where models might differ from hardware.

This section contains the following subsections:

- *1.3.1 How accurate are Fast Models?* on page 1-15.
- *1.3.2 Timing accuracy of Fast Models* on page 1-15.
- *1.3.3 Bus traffic in Fast Models* on page 1-15.
- *1.3.4 Instruction prefetch in Fast Models* on page 1-16.
- *1.3.5 Out-of-order execution and write-buffers in Fast Models* on page 1-17.
- *1.3.6 Caches in Fast Models* on page 1-17.
- *1.3.7 Global exclusive monitor in Fast Models* on page 1-18.

1.3.1 How accurate are Fast Models?

Fast Models aim to be accurate to the view of the system programmer. Architecturally correct software cannot tell between running on hardware and running on the model.

Software is able to detect differences between hardware and the model, but these differences generally depend on behavior that is not precisely specified. For example, it is possible to detect differences in the exact timings of instructions and bus transactions, effects of speculative prefetch and cache victim selection. Certain classes of behavior are specified as unpredictable and these cases are detectable by software. A program that relies on such behavior, even unintentionally, is not guaranteed to work reliably on any device, or on a Fast Model. Programs that exploit this behavior might execute differently between the hardware and the model.

Fast Models are only concerned with accuracy from the point of view of the program running on the processors. They do not attempt to accurately model bus transactions. PVBus provides the infrastructure to ensure that the program gets the correct results.

1.3.2 Timing accuracy of Fast Models

Fast Models do not model instruction timing accurately. The simulation as a whole has a very accurate concept of timing, but the *Code Translation* (CT) processors do not claim to dispatch instructions with device-like timing.

In general, a processor issues a set of instructions (a “quantum”) at the same point in simulation time, and then waits for some amount of time before executing the next quantum. The timing is arranged so that the processor averages one instruction per clock tick.

The consequences of this are:

- The perceived performance of software running on the model differs from real-world software. (In particular, memory accesses and arithmetic operations all take a significant amount of time.)
- A program might be able to detect the quantized execution behavior of a processor, for example by polling a high-resolution timer.
- All instructions in a quantum are effectively atomic.

  ——— **Note** ———

  This might mask some race-condition bugs in software.

  ———————————

1.3.3 Bus traffic in Fast Models

PVBus can simulate the behavior of individual bus transactions passing through a hierarchy of bus fabric, but it employs several techniques to optimize this process.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

1. PVBus generally decodes the path between a bus master and the bus slave the first time a transaction is issued. All subsequent transactions to the same address are automatically sent to the same slave, without passing through the intervening fabric.
2. For accesses to normal memory, the master can cache a pointer to the (host) storage that holds the data contents of the memory. The master can read and write directly to this memory without generating bus-transactions.
3. For instruction-fetch, and for operations such as repeated DMA from framebuffer memory, PVBus provides an optimization called “snooping”, that informs the master if anyone else could have modified the contents of memory. If no changes have occurred the master can avoid the need to re-read memory contents.

If a piece of bus fabric wants to intercept and log all bus transactions, it can defeat these optimizations by claiming to be a slave device. It can then log all transactions and can reissue identical transactions on its own master port. However, doing this slows all bus transactions and significantly impacts simulation performance.

Note

If direct accesses to memory by the CT engine are intercepted by the fabric, the processor is forced to single step. Execution is much slower than normal operation with translated code.

The bus traffic generated by a processor is not representative of real traffic:

Timing differences
Re-ordering and buffering of memory accesses, out-of-order execution, speculative prefetch and drain-buffers can cause timing differences. They are not modeled, since they are not visible to the programmer except in situations where a cluster program contains race conditions that violate serial-consistency expectations.

Bus contention
Fast Models do not model the time taken for a bus transaction, so they cannot model the effects of multiple transactions contending for bus availability.

Size of access
Fast Models do not attempt to generate the same types of burst transaction from the processor for accesses to multiple consecutive locations.

Instruction fetch
The behavior of the instruction prefetch unit of a processor is not modeled to match the hardware implementation.

Behavioral differences
In some software, the trace of instruction execution is dependent on timing effects. For example, if a loop polls a device waiting for a 10ms time-out, the number of iterations of the polling loop depends on the rate of instruction execution.

Related references

1.3.4 Instruction prefetch in Fast Models on page 1-16

1.3.4 Instruction prefetch in Fast Models

The CT engine in the processor models relies on Fast Models PVBus optimizations. It only performs code-translation if it has been able to prefetch and snoop the underlying memory. It then need not issue bus transactions until the snoop handling detects an external modification to memory.

If the CT engine cannot get prefetch access to memory, it drops to single-stepping. This single-stepping is very slow (~1000x slower than translated code execution).

Real processors attempt to prefetch instructions ahead of execution and predict branch destinations to keep the prefetch queue full. The instruction prefetch behavior of a processor can be observed by a program that writes into its own prefetch queue (without using explicit barriers). The architecture does not define the results.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

The CT engine processes code in blocks. The effect is as if the processor filled its prefetch queue with a block of instructions, then executed the block to completion. As a result, this virtual prefetch queue is sometimes larger and sometimes smaller than the corresponding hardware. In the current implementation, the virtual prefetch queue can follow small forward branches.

With an L1 instruction cache turned on, the instruction block size is limited to a single cache-line. The processor ensures that a line is present in the cache at the point where it starts executing instructions from that line.

In real hardware, the effects of the instruction prefetch queue are to cause additional fetch transactions. Some of these are redundant because of incorrect branch prediction. This causes extra cache and bus pressure.

Related references

1.3.3 Bus traffic in Fast Models on page 1-15

1.3.5 Out-of-order execution and write-buffers in Fast Models

Hardware memory is Weakly Ordered, but Fast Models memory is Strongly Ordered.

The CT implementation executes instructions sequentially. One instruction is retired before the next starts to execute. In a real processor, multiple memory accesses can be outstanding, and can complete in a different order from their program order. Writes can also be delayed in a write-buffer.

The programmer visible effects of these behaviors is defined in the architecture as the Weakly Ordered memory model, which the programmer must be aware of when writing lock-free cluster code.

Within Fast Models, memory accesses happen in program order, effectively as if all memory is Strongly Ordered.

1.3.6 Caches in Fast Models

Fast Models with cache-state modeling enabled can replicate some types of failure-case, but not all types.

The effects of caches are programmer visible because they can cause a single memory location to exist as multiple inconsistent copies. If caches are not correctly maintained, reads can observe stale copies of locations, and flushes/cleans can cause writes to be lost.

There are three ways in which incorrect cache maintenance can be programmer visible:

From the D-side interface of a single processor
: The only way of detecting the presence of caches is to create aliases in the memory map, so that the same range of physical addresses can be observed as both cached and non-cached memory.

From the D-side of a single processor to its I-side
: Stale instruction data can be fetched when new instructions have been written by the D-side. This can either be because of deliberate self-modifying code, or as a consequence of incorrect OS demand paging.

Between one processor and another device
: For example, another processor in a non-coherent MP system, or an external DMA device.

Fast Models with cache-state modeling enabled can replicate all of these failure-cases. However, they do not attempt to reproduce the following effects of caches:

- Changes to timing behavior of a program because of cache hits/misses (because the timing of memory accesses is not modeled).
- Ordering of line-fill and eviction operations.
- Cache usage statistics (because the models do not generate accurate bus traffic).
- Device-accurate allocation of cache victim lines (which is not possible without accurate bus traffic modeling).
- Write-streaming behavior where a cache spots patterns of writes to consecutive addresses and automatically turns off the write-allocation policy.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

The Cortex®-A9 and Cortex-A5 models do not model device-accurate MESI behavior. The Cortex-A15 and Cortex-A7 models do simulate hardware MOESI state handling, and can handle cache-to-cache snoops. In addition, they model the AMBA® 4 ACE cache-coherency protocols over their PVBus ports, so can be connected to a model of an ACE Coherent Interconnect (such as the CCI-400 model) to simulate coherent sharing of cache contents between processors.

It is not possible to insert devices between the processor and its L1 caches. In particular, you cannot model L1 traffic, although you can tell the model not to model the state of L1 caches.

1.3.7 Global exclusive monitor in Fast Models

This section describes the behavior of the global exclusive monitor, when `cache-state-modelled` is `false` and when it is `true`.

Whether `cache-state-modelled` is `false` or `true`, when the system enables the data caches, they implement the monitor for cacheable nonshared and shared memory using the cache coherency protocol. The monitor includes other clusters that are connected with a cache coherent interconnect. Most global monitor behavior, like exclusive granule size, is the same whether `cache-state-modelled` is `false` or `true`. In particular, the data caches follow behavior that the Arm architecture prescribes in both cases. However, some behaviors do rely on the full modeling of cache functionality. An example is that the clearing of a monitor as a result of a cache line eviction does not happen with `cache-state-modelled` as `false`.

The RAMDevice component and PVBus to AMBAPV bridges do not contain global monitors. As a result, exclusive stores that reach a RAMDevice fail unless they go through an exclusive monitor. Some platforms might need a global monitor outside of the cache coherency domains. These platforms must include a system level monitor in the same place in the bus hierarchy as in the hardware. See the FVP_VE and FVP_Base example platforms for examples of how to use the PVBusExclusiveMonitor component.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

1.4 Processor implementation

This section outlines the differences between the code translation models and the hardware.

This section contains the following subsections:

- *1.4.1 Mapping PMU events to MTI trace sources* on page 1-19.
- *1.4.2 Caches in PV models* on page 1-20.
- *1.4.3 GICv3 in PV models* on page 1-22.
- *1.4.4 CP14 Debug coprocessor* on page 1-22.
- *1.4.5 MicroTLBs* on page 1-22.
- *1.4.6 TLBs in PV models* on page 1-22.
- *1.4.7 Memory access in PV models* on page 1-23.
- *1.4.8 Timing in PV models* on page 1-23.
- *1.4.9 VIC ports in PV models* on page 1-24.

1.4.1 Mapping PMU events to MTI trace sources

In Fast Models, many PMU events are modeled using MTI trace sources. These events are exposed through the standard PMU interface. The user does not need to interface with MTI.

The following tables show which PMUv1 and PMUv2 events are modeled, and which MTI trace sources are used to model them. The associated event numbers are as listed in the Arm Architecture Reference Manual Armv7-A and Armv7-R edition, section C12.8.2 *Common event numbers*.

Table 1-1 PMUv1 event mappings

PMUv1 event	MTI trace sources	Notes
Software increment (`0x00`)	-	Modeled, but not using MTI.
Level 1 instruction cache refill (`0x01`)	`ALLOC_LINEFILL`	-
Level 1 instruction TLB refill (`0x02`)	`MMU_TLB_FILL`	-
Level 1 data cache refill (`0x03`)	`ALLOC_LINEFILL`	-
Level 1 data cache access (`0x04`)	`CACHE_READ_HIT`, `CACHE_READ_MISS` `CACHE_WRITE_HIT`, `CACHE_WRITE_MISS`	-
Level 1 data TLB refill (`0x05`)	`MMU_TLB_FILL`	-
Load (`0x06`)	`CORE_LOADS`, `NV2_CORE_LOADS`, `MEMTAG_LOAD_INST`, `SVE_LD_RETIRED`	-
Store (`0x07`)	`CORE_STORES`, `NV2_CORE_STORES`, `MEMTAG_STORE_INST`, `SVE_ST_RETIRED`	-
Instruction architecturally executed (`0x08`)	-	Modeled, but not using MTI.
Exception taken (`0x09`)	`EXCEPTION`	-
Exception return (`0x0A`)	`EXCEPTION_RETURN_PREBRANCH`	-
Write to `CONTEXTIDR` (`0x0B`)	`CONTEXTIDR`	-
Software change of the PC (`0x0C`)	`BRA_DIR`, `BRA_INDIR`.	-
Immediate branch (`0x0D`)	`BRA_DIR`, `BRA_DIR_FAIL`.	-
Procedure return (`0x0E`)	`BRA_RET`	-

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 1-1 PMUv1 event mappings (continued)

PMUv1 event	MTI trace sources	Notes
Unaligned load or store (0x0F)	UNALIGNED_LDST_RETIRED, SVE_UNALIGNED_LDST_RETIRED	-
Mispredicted or not predicted branch speculatively executed (0x10)	BRANCH_MISPREDICT	Only if the BranchPrediction plug-in is loaded.
Cycle count (0x11)	-	Modeled, but not using MTI.
Predictable branch speculatively executed (0x12)	-	Not modeled.

Table 1-2 PMUv2 event mappings

PMUv2 event	MTI source	Notes
Data memory access (0x13)	-	Not modeled.
Level 1 instruction cache access (0x14)	CACHE_READ_HIT, CACHE_READ_MISS	-
Level 1 data cache write-back (0x15)	-	Not modeled.
Level 2 data cache access (0x16)	CACHE_READ_HIT, CACHE_READ_MISS, CACHE_WRITE_HIT, CACHE_WRITE_MISS	-
Level 2 data cache refill (0x17)	ALLOC_LINEFILL	
Level 2 data cache write-back (0x18)	-	Not modeled.
Bus access (0x19)	CORE_LOADS, CORE_STORES, MEMTAG_LOADS, MEMTAG_STORES	-
Memory error (0x1A)	-	Not modeled.
Instruction speculatively executed (0x1B)	-	Not modeled.
Write to TTBR (0x1C)	TTBR_WRITE	-
Bus cycle (0x1D)	INST	This value is an approximation.

Related information

Arm Architecture Reference Manual ARMv7-A and ARMv7-R edition

1.4.2 Caches in PV models

Some processor models have PV-accurate caches, but others do not model Level 1 or Level 2 caches.

Cores that have PV-accurate cache implementation provide a functionally-accurate model. For more information, see the processor component descriptions.

Other PV models do not model Level 1 or Level 2 caches. The system coprocessor registers related to cache operations permit cache aware software to work, but in most cases they only check register access permissions.

The registers affected on all code translation processor models are:

- Invalidate and/or Clean Entire ICache/DCache.
- Invalidate and/or Clean ICache/DCache by MVA.
- Invalidate and/or Clean ICache/DCache by Index.
- Invalidate and/or Clean Both Caches.
- Cache Dirty Status.
- Data Write Barrier.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

- Data Memory Barrier.
- Prefetch ICache Line.
- ICache/DCache lockdown.
- ICache/DCache master valid.

Functional caches in Fast Models - about

Fast Models implement two types of cache model: register model and functional model.

A register model provides all the cache control registers so that cache operations succeed, but does not model the state of the cache. All accesses go to memory.

A functional model tracks cache state and its contents at each level of the memory hierarchy. Incorrect cache management might return incorrect data, as it would on real hardware.

Fast Models provide:
- System IPs that support caches.
- Register models of caches on all processors that support caches and also the PL310 cache controller (PL310_L2CC).
- Functional models of caches integrated into Cortex cores.

For a core with no L2 cache, the configuration parameters are:

`icache-state-modelled`
Set whether the I-cache has stateful implementation.

`dcache-state-modelled`
Set whether the D-cache has stateful implementation.

For Armv7-A cores with an L2 cache, the configuration parameters are:

`l1_icache-state-modelled`
Set whether L1 I-cache has stateful implementation.

`l1_dcache-state-modelled`
Set whether L1 D-cache has stateful implementation.

`l2_cache-state-modelled`
Enable unified Level 2 cache state model.

For Armv8-A cores with an L2 cache, the configuration parameters are:

`icache-state_modelled`
Set whether L1 I-cache has stateful implementation. Instructions at L2 or L3 are not cached in Fast Models.

`dcache-state_modelled`
Set whether D-cache has stateful implementation at L1, L2, and L3.

Functional caches in Fast Models - performance

Enabling functional cache modeling is likely to reduce performance.

Enable the L1 and L2 functional caches together. For consistent system operation, ARM recommends that you either disable functional behavior completely or enable it for both I and D L1 caches and the L2 cache.

Cache enablement must be system wide. If you enable cache state modeling in any component then you must enable it in all components in the system, including all cores (L1 and L2) and any external cache controller (such as the PL310_L2CC) and any interconnect that has caches.

If platform memory is being modeled outside of the Fast Models environment, for example in a SystemC environment, use of functional cache modeling might improve performance if there is no other fast route to memory.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

1.4.3 GICv3 in PV models

This section describes the model of the GICv3.

GICv3 - about

The PV models implement the *Generic Interrupt Controller architecture version 3* (GICv3).

The GICv3 architecture defines two parts: the core interface (integrated into the core) and the *Interrupt Redistribution Infrastructure* (IRI). You can configure all ARMv8-A cores to include a GICv3 interface. You can integrate a separate GICv3IRI component into your platforms. Communication between the core and the IRI is over an architected packet interface. An internal communication protocol represents the packets that pass over this interface.

You can configure the GICv3 models in some platforms to act as though they were GICv2 or GICv2-M models. Even in this mode, you need a GICv3IRI component and supported core. Configure them to comply with the same standard.

Related references

3.9.29 GICv3IRI on page 3-746

GICv3 - functionality

Some features differ in the model.

- Support of the GITS_CTLR.Quiesce bit is not complete.
- Support of ITS save/restore is not complete. Configuration stays within the model and it does not use allocated memory.
- GICD_CTLR.RWP does not perform adequately. This difference is only an issue if you use the distributor in systems with delaying interface between the distributor and the cores. DO NOT use this version of the model for simulation of the GIC in a setup where interfaces are not instantly reactive.
- Set the environment variable `FASTSIM_GIC_MEMORY_MAP` to 1 so the memory map of certain models that are included in the platform being run will be printed to `stderr`. This functionality is available for all GICv3 and later models.

GICv4 - functionality

GICv4 is an extension of the GICv3 architecture. It allows the direct injection of LPIs into a virtualized system through the `virtual-lpi-support` parameter of the `GICv3IRI` or `GICv3IRI_Filter` component.

In addition to requiring the presence of an ITS that is configured as shown in *3.9.29 GICv3IRI* on page 3-746, GICv4 requires you to enable the virtual LPIs feature and to configure a virtual PE table using the parameters shown in the following example:

```
"virtual-lpi-support"=true,
"GITS_BASER4-type"=2 //Type 2 is Virtual PEs.
                     //Such a table is needed for GICv4 functionality.
```

1.4.4 CP14 Debug coprocessor

Some models fully implement the CP14 Debug coprocessor registers. Other models only implement the DSCR register. This register reads as 0 and ignores writes.

External debugging must be used to debug systems containing PV models.

1.4.5 MicroTLBs

The models do not implement device accurate MicroTLBs, or system coprocessor registers related to MicroTLB state.

1.4.6 TLBs in PV models

The PV models implement *Translation Lookaside Buffers* (TLBs) and model most aspects of TLB behavior.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Note

If the `device-accurate-tlb` parameter is set to `false`, the simulation uses a different number of TLBs if this improves simulation performance. The simulation is architecturally accurate, but not device accurate. Architectural accuracy is almost always sufficient. Set `device-accurate-tlb` parameter to `true` if you require device accuracy.

These TLB registers do not have working implementations:

- Primary memory remap register.
- Normal memory remap register.

In addition, the simulation does not distinguish peripheral accesses from data accesses, so it ignores configuration of the peripheral port memory remap register.

1.4.7 Memory access in PV models

This section describes the kinds of memory access in PV models.

About memory access in PV models

PV models use a PVBusMaster subcomponent to communicate with slaves in a System Canvas generated system. This provides efficient access to memory-like slaves and relatively efficient access to device-like slaves.

Memory access in PV models differs from real hardware to enable fast modeling of the processor:

- All memory accesses are performed in programmer view order.
- Unaligned accesses, where permitted, are always performed as byte transfers.

In addition, some PV models do not use all the transaction states available in a PVBus transaction. The Privileged and Instruction flags are set correctly for ARMv7 processors but might not be set correctly in earlier architectures. However all memory accesses are atomic so SWP instructions behave as expected.

I-side access in PV models

PV models cache translations of instructions fetched from memory-like slaves. The models might not perform further access to those slaves for significant periods. A slave can force the model to reread the memory by declaring that the memory has changed.

PV models do not model a prefetch queue but the code translation mechanism effectively acts as a prefetch queue of variable depth. ARM recommends that you follow the standards in the *Arm® architecture specifications* for dealing with prefetch issues, such as self modifying code, and use appropriate cache flushing and synchronization barriers.

Translation of instructions only occurs for memory-like slaves, which are those declared by devices as having type `pv::MEMORY`. Instructions fetched from device-like slaves are repeatedly fetched, decoded and executed, significantly slowing down model performance.

D-side access in PV models

PV models cache references to the underlying memory of memory-like slaves, and might not perform further accesses to those slaves over the bus for significant periods.

Slaves declared as type `pv::MEMORY` provide the fastest possible memory access for PV processors.

Slaves declared as type `pv::DEVICE` are normally used for peripheral access.

1.4.8 Timing in PV models

Programmer's View (PV) models are loosely timed.

- Caches and write buffers are not modeled, so all memory access timing is effectively zero wait state.
- All instructions execute, in aggregate, in one cycle of the component master clock input.
- Interrupts are not taken at every instruction boundary.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

- Some sequences of instructions are executed atomically, ahead of the master clock of a component, so that system time does not advance during execution. This difference in behavior can affect sequential access of device registers, where devices are expecting time to move on between accesses.
- DMA to and from *Tightly Coupled Memory* (TCM) is atomic.

1.4.9 VIC ports in PV models

The ARMCortexR4CT and ARMCortexR5x*n*CT models implement a simplified model of the *Vectored Interrupt Controller* (VIC) port.

The protocol consists of two ports:

- The vic_addr port signals the vectored interrupt address from the external VIC.
- The vic_ack port signals the VIC that an interrupt has been detected and is being serviced.

The expected interrupt sequence is:

1. The software enables the VIC interface by setting the VE bit in the CP15 control register and setting up suitable interrupt routines.
2. The VIC asserts IRQ.
3. Some time later, the processor detects and responds to the IRQ by asserting vic_ack.
4. The VIC writes the vector address to the processor using vic_addr.
5. The processor de-asserts vic_ack.
6. The processor transfers control to the vectored address returned from the VIC.

The interaction between the processor and the VIC is untimed after the processor acknowledges the interrupt, so certain interrupt sequences cannot occur in the code translation processor models.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

1.5 Fast Models CADI implementation

Fast Models implement a subset of CADI functionality.

The following CADI methods are not supported by Fast Models:

CADI class:

Register API

`CADIGetCommitedPCs()`

Memory API

`CADIMemGetOverlays()`

Virtual Memory API

`PhysicalToVirtual()`

Cache API

`CADIGetCacheInfo()`
`CADICacheRead()`
`CADICacheWrite()`

Execution API

`CADIExecLoadApplication()` - Implemented by CPU components only.
`CADIExecUnloadApplication()`
`CADIExecGetLoadedApplications()` - Implemented by CPU components only. It only returns the most recently loaded application if more than one have been loaded.
`CADIExecAssertException()`
`CADIExecGetPipeStages()`
`CADIExecGetPipeStageFields()`
`CADIGetCycleCount()`

CADICallbackObj class:

- `appliOpen()`
- `appliClose()`
- `cycleTick()`
- `killInterface()`

CADIDisassembler class:

- `GetSourceReferenceForAddress()`
- `GetAddressForSourceReference()`
- `GetInstructionType()`

CADIDisassemblerCB class:

- `ReceiveSourceReference()`

The following tables describe the implementation of other CADI functionality by processor models:

Table 1-3 CADI broker implementation

Feature	Implemented	Remarks
Factories	Yes	Model DLL provides a single factory.
Instances	Yes	Factory can only instantiate one model at a time.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 1-4 CADI target implementation

Feature	Implemented	Remarks
# breakpoints	Yes	No intrinsic limit.
Breakpoint disabling	Yes	You can disable global breakpoints, but not user breakpoints set through `CADIBptSet()`. Debuggers usually disable breakpoints with `CADIBptClear()` and re-enable them with `CADIBptSet()`, so this does not affect debugger GUIs, but it does affect the CADI C++ interface. Global breakpoints are built-in and permanent, so `CADIBptSet()/Clear()` cannot set or delete them. With `CADIBptConfigure()`, you can only enable/disable these global breakpoints to stop on certain processor events.
Breakpoint ignore count	No	-
Breakpoint formal conditions	No	-
Breakpoint free-form conditions	No	-
Breakpoint setting	Yes[a]	-
Breakpoint types	Yes: `CADI_BPT_MEMORY`, `CADI_BPT_PROGRAM`, `CADI_BPT_PROGRAM_RANGE`	All registers in the Vector register group have disabled register breakpoints. You cannot create new register breakpoints.
Breakpoint triggers	Yes: `CADI_BPT_TRIGGER_ON_MODIFY`, `CADI_BPT_TRIGGER_ON_READ`, `CADI_BPT_TRIGGER_ON_WRITE`	`CADI_BPT_TRIGGER_ON_WRITE` approximates `CADI_BPT_TRIGGER_ON_MODIFY`. If either is set, the simulation stops when a value is written, even when it does not change.
`CADIExecSingleStep()`	Yes	Fast Models ignores the `stepCycle` and `stepOver` arguments.
Intrusive debug	Yes	Stop/start can affect processor scheduling: single stepping and multiple stepping both change scheduling, and so are intrusive.
Memory read/write	Yes[a]	-
Reset	Yes	One reset level is supported. When calling `CADIExecReset()`, Arm recommends the simulation should use a syncLevel of 1 or higher for best results.
Runnable	Yes	-
Stop simulation	Yes	-

[a] Not permitted while target is running.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

1.6 CADI interactions with processor behavior

Architecturally, M series processors have different reset behavior to that of A and R series processors.

For both hardware and model processors, a reset consists of asserting, and then deasserting the reset pin. However:

M series
: The architecture documentation specifies that on asserting the reset pin, the processor stops executing instructions. On deasserting the reset pin, the VTOR is given its reset value, SP_main is read from address VTOR+0, and the PC is read from address VTOR+4. See the Reset behavior section and the *Vector Table Offset Register*, VTOR section in the *ARMv7-M Architecture Reference Manual*. Also, the processor begins fetching and executing instructions.

A and R series
: Asserting the reset pin causes the processor to stop executing instructions and the PC to be given its reset value, which depends on VINITHI, and the SP is `UNKNOWN`. See the Reset section in the *ARM Architecture Reference Manual ARMv7-A and ARMv7-R edition*. On deasserting the reset pin, the processor begins fetching and executing instructions.

The behavior of reset on M series processors interacts differently with CADI debugging functionality than it does for A and R series processors.

On A and R series processors, after the reset pin of the processor is asserted, a new application can be loaded using the `CADIExecLoadApplication()` call in CADI. This loads an application into memory, and sets the PC if a start address is specified in the application. When reset is deasserted, the processor begins fetching and executing from the start address as expected. Similarly, if a debugger asserts reset on the processor, it can modify memory with a sequence of `CADIMemWrite()` calls, update the PC with a `CADIRegWrite()` call, and when reset is deasserted the processor begins to fetch and execute from the PC.

On M series processors, these techniques do not work because after reset is deasserted the processor updates SP_main and the PC, overwriting the settings made by the CADI calls.

For these techniques to be possible, the M series processor models differ slightly from the architectural reset behavior. When reset is asserted, the M series makes note of whether the PC or SP are modified before reset is next deasserted. If there are any CADI writes to the PC or SP registers, either directly through `CADIRegWrite` or indirectly through `CADIExecLoadApplication()`, this is tracked. When reset is deasserted, if the PC has been modified using CADI, the PC retains the value written to it, otherwise it reads it from address VTOR+4. Similarly, if the SP has been modified using CADI, SP_main retains the value written to it, otherwise it reads it from address VTOR+0.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

1.7 CADI sync watchpoints

The CADI/*Synchronous CADI* (SCADI) interfaces of processor components support watchpoints. Memory components like RAMDevice do not.

When the simulator hits a watchpoint, the CADI/SCADI interface emits a sequence, typically only one, of `modeChange(CADI_EXECMODE_Bpt, bptNumber)` callbacks, where `bptNumber` is the breakpoint ID of the watchpoint. After this sequence it sends a `modeChange(CADI_EXECMODE_Stop)` callback. Only after the debugger receives this `Stop` callback might it inspect the state of the model.

You can read additional information about the last of the hit watchpoints from these CADI registers declared by the processor in register group `Simulation`:

`memoryBptPC`
The PC of the instruction that caused the watchpoint to hit.

`memoryBptAccessVA`
Virtual address of the access or sub-access that caused the watchpoint to hit.

`memoryBptAccessSize`
Size in bytes of the access or sub-access that caused the watchpoint to hit.

`memoryBptAccessRW`
Type of access or sub-access that caused the watchpoint to hit: 1 = read, 2 = write, 3 = both, 0 = no access.

These registers are not memory (or CPnn-) mapped anywhere and are not accessible to target programs. These registers are read-only. `memoryBptPC` and `memoryBptAccessVA` are 32 or 64 bit depending on the architecture. `memoryBptAccessSize` and `memoryBptAccessRW` are 32 bit.

When multiple accesses have hit watchpoints at the same time, for example during the same instruction, the information contained in these registers is valid and consistent only for one of these accesses.

Whenever at least one CADI watchpoint is set on a processor, the syncLevel on that processor is at least 2.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

1.8 Non-CADI sync watchpoints

This section describes syncLevel enum values, semantics, and behaviors, with syncLevel values listed from fast to slow simulation, from fewer to more requirements.

This section contains the following subsections:

- *1.8.1 syncLevel definitions* on page 1-29.
- *1.8.2 Controlling and observing the syncLevel* on page 1-30.

1.8.1 syncLevel definitions

Definitions for 0 - OFF, 1 - SYNC_STATE, 2 - POST_INSN_IO, and 3 - POST_INSN_ALL.

`syncLevel 0: OFF`

- The simulator runs as fast as possible. It does not permit inspection of the processor registers while the simulation is running, and does not stop synchronously when requested to do so.
- After enabling or disabling a trace source, it is undefined how many instructions will be executed before the change takes effect.
- Quantum end detection guarantees that a quantum is not overshot indefinitely. Quantum end detection applies to (but is not limited to) backward branches, indirect jumps, exceptions, and atomic operation retries. In addition to temporal quantum end detection, some events may end a quantum, like executing a barrier, entering a low power state, or accessing a peripheral. Target software and simulation controllers must not rely on a specific scheduling pattern based on these quantum end check points.
- Use cases: normal fast simulation and normal CADI debugging while no CADI watchpoint is set.

——— **Note** ———

When using CADI reset on a model, for best results, Arm recommends using a syncLevel of 1 or higher.

———————————

`syncLevel 1: SYNC_STATE`

- The simulation runs slightly slower than syncLevel 0. SCADI can read the up-to-date values of the processor registers, including PC and instruction count. You cannot stop the simulation synchronously.
- After enabling or disabling a trace source, it is undefined how many instructions will be executed before the change takes effect.
- Quantum end detection is as for syncLevel 0.
- Use cases: external breakpoints that block the simulation, inspect state of processor/memory from within a peripheral or memory access.

`syncLevel 2: POST_INSN_IO`

- As for syncLevel 1, except that you can stop the simulation synchronously from within all LD/ST and similar instructions. The simulation stops immediately after the current LD/ST instruction has been completely executed (post instruction).
- After enabling or disabling a trace source, it is likely, but not guaranteed, that the change will be visible sooner than with syncLevel 0 or 1.
- Quantum end detection is as for syncLevel 1, plus it includes the end of LD/ST instructions.
- Use cases: CADI watchpoints, external breakpoints, stopping from within LD/ST-related MTI callbacks.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

syncLevel 3: POST_INSN_ALL

- As for syncLevel 2, except that you can stop the simulation synchronously from within any instruction. The simulation stops immediately after the current instruction has been completely executed (post instruction).
- After enabling or disabling a trace source, the change will be visible at the next instruction that is executed.
- Quantum end detection is as for syncLevel 2. This allows switching between syncLevels 2 and 3 without changing the simulation scheduling.
- Use cases: a `Stop` from within arbitrary MTI callbacks such as the `INST` callback. This syncLevel is a fallback for all use cases that do not fall into syncLevels 0-2.

1.8.2 Controlling and observing the syncLevel

CADI watchpoints automatically register and unregister for their required syncLevel (`POST_INSN_IO`). All other use cases must explicitly register and unregister for the syncLevel they require.

Users of syncLevel write to a set of non-architectural processor registers in the CADI and SCADI interface to register and unregister for specific syncLevels. Processor registers are more suitable than CADI parameters for exposing an interface that has side effects on writes and where values might change spontaneously.

This is the exposed interface to control and observe the value of syncLevel. All these registers are in the CADI/SCADI register group `Simulation` for each CT processor that contains non-architectural, simulator-specific registers. All are 32-bit integer registers. Users of syncLevel write to these registers to register and unregister for the syncLevel they require:

syncLevelSyncStateRegister
Users write to this register for `SYNC_STATE`. Write-only.

syncLevelSyncStateUnregister
Users write this to unregister for `SYNC_STATE`. Write-only.

syncLevelPostInsnIORegister
Users write to this register for `POST_INSN_LDST`. Write-only.

syncLevelPostInsnIOUnregister
Users write this to unregister for `POST_INSN_LDST`. Write-only.

syncLevelPostInsnAllRegister
Users write this to register for `POST_INSN_ALL`. Write-only.

syncLevelPostInsnAllUnregister
Users write this to unregister for `POST_INSN_ALL`. Write-only.

These registers are only for debugging and visibility in the debugger, and syncLevel users do not usually access them at all:

syncLevel
Current syncLevel. Read-only.

syncLevelSyncStateCount
User counter. Read/write (use as read-only).

syncLevelPostInsnIOCount
User counter. Read/write (use as read-only).

syncLevelPostInsnAllCount
User counter. Read/write (use as read-only).

minSyncLevel
Same as the `min_sync_level` parameter, described below. Read/write.

The `*Register` and `*Unregister` registers are for syncLevel users to register and unregister themselves, by writing the value 0 to them. Changes to the syncLevel become effective at the next stop event

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

checkpoint. In addition, syncLevel users can write to these registers any time before the simulation is running, for example from the `init()` simulation phase. The change takes effect immediately when the simulation is run.

The other registers are only present to make the debugging of these mechanisms and their users easier. The `syncLevel` register enables you to see what kind of performance you can expect from the model. You must treat access to these other registers as read-only. You can write to them, however, to permit debugging the syncLevel mechanisms.

These registers are not memory (or CPnn-) mapped anywhere, and are not accessible to target programs.

In addition to this debug register interface, there is a CADI parameter that can influence the syncLevel:

```
min_sync_level (default=0, type=int, runtime=true)
```

This parameter enables you to control the minimum syncLevel by the CADI parameter interface. This is not the intended primary interface to control the syncLevel because it does not enable multiple independent syncLevel users to indicate their requirements to the simulator. It is primarily for debugging purposes and for situations where a single global syncLevel setting is sufficient. You can change this parameter at runtime, and changes become effective at the next stop event checkpoint. Reading this parameter value returns the `min_sync_level`, not the current syncLevel. This parameter is only an additional way of controlling the syncLevel and controls the same mechanisms as the register interface.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

1.9 SCADI

This section describes *Synchronous CADI* (SCADI) for components.

This section contains the following subsections:

- *1.9.1 About SCADI* on page 1-32.
- *1.9.2 Intended uses of CADI and SCADI* on page 1-32.
- *1.9.3 Responsibilities of the SCADI caller* on page 1-33.
- *1.9.4 SCADI interface access* on page 1-33.
- *1.9.5 SCADI semantics* on page 1-34.

1.9.1 About SCADI

A SCADI interface enables you to request a synchronous stop of the simulation. It offers direct synchronous, but unsynchronized, register and memory read access.

The intention is that this interface is used by code that is inherently synchronized with the simulation, in particular from code that is part of the simulation itself (simulation thread). Examples of code that could use this interface are SystemC and LISA peripheral device/bus access functions, and MTI callback functions that are always called synchronously by the simulation itself.

Note

The SCADI interface is not a replacement for CADI. It is an additional interface with clear semantics that differ slightly from CADI.

1.9.2 Intended uses of CADI and SCADI

Component Architecture Debug Interface (CADI) and *Synchronous CADI* (SCADI) have separate design principles.

CADI

is used by debuggers and simulation controllers to control the simulation. Execution control functions are always asynchronous and are usually called from a non-simulation thread, usually the debugger GUI thread. You can also use CADI to read/write registers, memory and so on, but only from non-simulation threads, for example only from the debugger thread.

SCADI

implements a specific subset of functions of CADI, that is, mainly read/write registers and memory, set and clear breakpoints, and get disassembly. You can only use SCADI from the simulation thread itself and from threads that can make sure on their own that the simulation is currently blocked, for example a debugger thread while it is sure that the simulation is in a stable state. SCADI is intended to be used from within peripheral read/write accesses while the simulation is running, or from within MTI callbacks that the simulation is running.

SCADI does not implement and execute control functions except `CADIExecStop()`, because only stop makes sense from within the simulation thread. Asynchronous functions such as `CADIExecContinue()` and `CADIExecSingleStep()` are not supported by SCADI, so for these you must use CADI instead. The `SCADI::CADIExecStop()` is the exception, because for CADI this means “stop when it is convenient for the simulation”, which is asynchronous, but `CADIExecStop()` is synchronous and means “stop now”, which you enable with the appropriate syncLevel > = 2.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

The guidelines are:

- Use CADI for execution control.
- Use CADI or SCADI for read/write registers and memory, depending on the situation:
  - Use SCADI if the caller can make sure that the accesses are (potentially inherently) synchronized with the simulation.
  - Use CADI if called from a debugger thread that does not know exactly whether the simulation is running or is in a stable state.

1.9.3 Responsibilities of the SCADI caller

Synchronous interface means that the caller of SCADI functions is always responsible for ensuring that the simulation is in a stable state.

Being in a stable state means that the simulation does not advance while the call is being made. The caller is also responsible for meeting all host thread and synchronization requirements of any external bus slaves that are directly or indirectly connected to the memory bus.

1.9.4 SCADI interface access

This section describes the `CAInterface::ObtainInterface()` method and how to use it.

About SCADI interface access

You can obtain the SCADI interface from the same CAInterface pointers that provide the CADI and the MTI interfaces.

This is done using the `CAInterface::ObtainInterface()` method. The interface names for the normal CADI and MTI interfaces are `CADI::IFNAME()` and `ComponentTraceInterface::IFNAME()`, respectively.

The interface name for the SCADI interface to pass into `ObtainInterface()` is `"eslapi.SCADI2"`, the interface revision is 0. There are no `IFNAME()` and `IFREVISION()` static functions defined for SCADI, and a plain string constant and integer constant (0) has to be used instead.

The pointer returned by `CAInterface::ObtainInterface("eslapi.SCADI2")` must be cast into an `eslapi::CADI` class pointer, that is, the same class as for `eslapi.CADI2`, the normal CADI interface class.

Access to SCADI from within LISA components

To access the SCADI interface of the LISA component itself, use the `getSCADI()` function. This returns the SCADI interface of the LISA component itself.

To access the SCADI interfaces of other components in the system, use the `getGlobalInterface()` function. This returns a CAInterface pointer to a simulation-wide Component Registry.

You then have to use this registry pointer with the `obtainComponentInterfacePointer()` function to obtain the SystemTraceInterface. This interface provides a list of all components in the system that provide trace data. You can use this list to access the SCADI interfaces of all the components.

Access to SCADI from within SystemC

To access the SCADI interfaces from within SystemC, use the `scx_get_global_interface()` function. This returns a CAInterface pointer to a simulation-wide Component Registry.

You then have to use this registry pointer with the `obtainComponentInterfacePointer()` function to obtain the SystemTraceInterface. This interface provides a list of all components in the system that provide trace data. You can use this list to access the SCADI interfaces of all the components.

Access to SCADI from MTI plug-ins

MTI plug-ins can gain access to SCADI interfaces by using the CAInterface pointer, which is passed as a parameter to the `RegisterSimulation()` function.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

This pointer has the same semantics as the CAInterface pointer returned by the `getGlobalInterface()` functions in the SystemC or LISA use cases.

Example code for retrieving a SCADI interface pointer of a component

This code demonstrates how to use `getGlobalInterface()` to retrieve a SCADI interface pointer of a specific component.

From within LISA, use `getGlobalInterface()`. From within SystemC, it is assumed that you have a pointer to the Fast Models platform called `platform`. From within an MTI plug-in, you do not have to call `getGlobalInterface()`, and you can use the CAInterface pointer passed to `RegisterSimulation()`.

Retrieving a SCADI interface pointer of a specific component

```
#include "sg/SGComponentRegistry.h"
eslapi::CADI *Peripheral::get_SCADI_interface(const char *instanceName)
{
  // get access to MTI interface (SystemTraceInterface)
  // - this code is for a SystemC peripheral
  // - for LISA this would be just 'getGlobalInterface()'
  // - for MTI plugins this would be just using the CAInterface *caif pointer passed
  //   into RegisterSimulation()
  eslapi::CAInterface *globalInterface = scx::scx_get_global_interface();
  const char *errorMessage = "(no error)";
  MTI::SystemTraceInterface *mti = 0;
  if (globalInterface)
     mti = sg::obtainComponentInterfacePointer<MTI::SystemTraceInterface>
                                        (globalInterface,
                                        "mtiRegistry",
                                        &errorMessage);
  if (mti == 0)
  {
     printf("ERROR:MTI interface not found! (%s)\n", errorMessage);
     return 0;
  }
  // find component with instanceName
  eslapi::CAInterface *compif = 0;
  for (MTI::SystemTraceInterface::TraceComponentIndex i = 0;
      i < mti->GetNumOfTraceComponents();
      i++)
  {
      const char *name = mti->GetComponentTracePath(i);
      if (verbose)
         printf("TEST MTI component '%s'\n", name);
      if (strcmp(name, instanceName) == 0)
         compif = mti->GetComponentTrace(i);
  }
  if (!compif)
  {
      printf("ERROR:instance '%s' not found while trying to get SCADI! \n",
              instanceName);
      return 0;
  }
  // get and return SCADI interface
  return static_cast<eslapi::CADI *>(compif->ObtainInterface("eslapi.SCADI2", 0, 0));
}
```

1.9.5 SCADI semantics

This section lists the available subset of functionality and the differences in semantics from the CADI interface.

The SCADI interface provides a subset of functionality of the CADI interface. All functions in SCADI differ from the CADI functions, that is, the caller is responsible for satisfying the constraints and requirements in terms of synchronization with the simulation.

These functions have the same semantics as in CADI:

- `CADIRegGetGroups()`.
- `CADIRegGetMap()`.
- `CADIRegGetCompount()`.
- `CADIMemGetSpaces()`.
- `CADIMemGetBlocks()`.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

- `CADIMemGetOverlays()`.
- `CADIGetParameters()`.
- `CADIGetParameterInfo()`.
- `CADIXfaceGetFeatures()`.

The following debug read access functions have the same semantics as in CADI, but the values that these read access functions return might differ from the values returned by the corresponding CADI functions because some of the accessed resource values might change during the execution of an instruction. The values of resources that are modified by the current instruction are UNKNOWN, except the PC, which always reflects the address of the last instruction that was issued. Reading and writing registers and memory while the simulation is running is generally only useful for `syncLevel >= SL_SYNC_STATE`. For SL_OFF, all registers and all memory locations are always UNKNOWN while the simulation is running.

- `CADIRegRead()`.
- `CADIMemRead()`.
- `CADIGetParameterValues()`.
- `CADIGetInstructionCount()`.
- `CADIGetPC()`.
- `CADIGetDisassembler()`.

The following write access functions always provide write access to the register in the `Simulation` register group. They also might provide limited write access semantics to certain resources. A SCADI implementation might choose not to support any write access to core registers and memory:

- `CADIRegWrite()`.
- `CADIMemWrite()`.
- `CADISetParameters()`.

The breakpoint functions have nearly all the same semantics as in the CADI interface, except that changes to breakpoints only become effective at the next stop event checkpoint defined by the current syncLevel. Changes might also only become visible in `CADIBptRead()` and `CADIBptGetList()` at this next synchronization point.

- `CADIBptGetList()`.
- `CADIBptRead()`.
- `CADIBptSet()`.
- `CADIBptClear()`.
- `CADIBptConfigure()`.

Execution control is limited to stopping the simulation. The simulation stops at the next synchronization point as defined by the current syncLevel:

- `CADIExecStop()`.

All other functions are not supported, and return `CADI_STATUS_CmdNotSupported`.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

1.10 Using the TelnetTerminal

This virtual component permits the transfer of UART data between a TCP/IP socket on the host and a serial port on the target.

The following figure shows a block diagram of one possible relationship between the target and host through the Terminal component. The TelnetTerminal block is what you configure when you define Terminal component parameters. The Virtual Machine is your FVP.

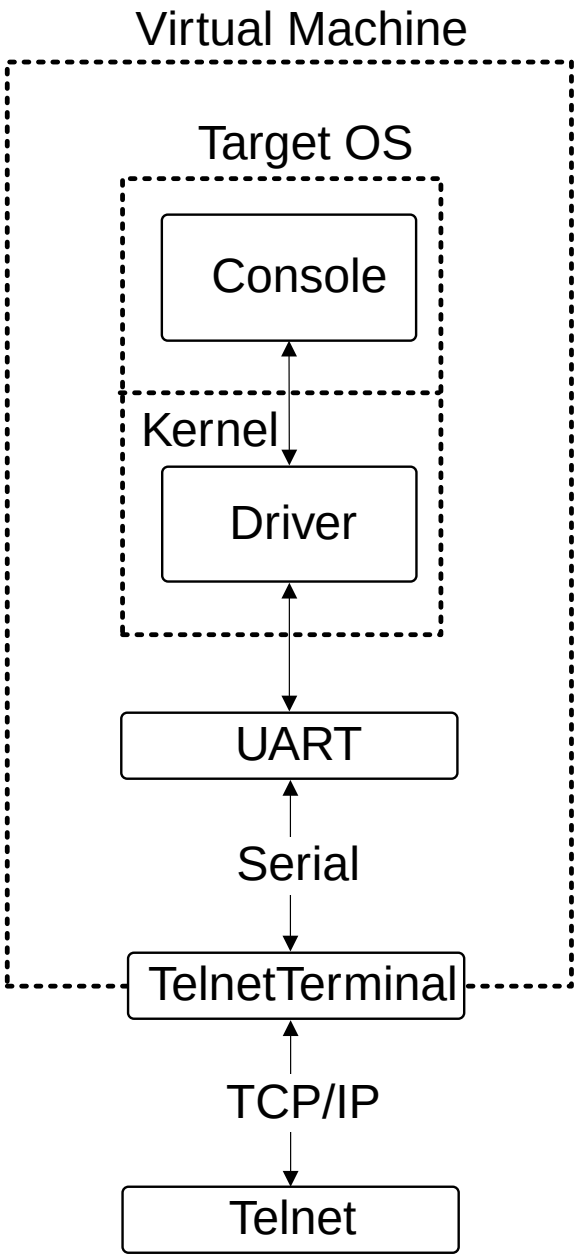

Figure 1-1 Terminal block diagram

On the target side, the console process that is invoked by your target OS relies on a suitable driver being present. Such drivers are normally part of the OS kernel. The driver passes serial data through a UART.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

The data is forwarded to the TelnetTerminal component, which exposes a TCP/IP port to the world outside of the model. This port can be connected to by, for example, a Telnet process on the host.

You can change the startup behavior for each of up to four TelnetTerminals by modifying the corresponding component parameters.

If the TelnetTerminal connection is broken, for example by closing a client telnet session, the port is re-opened on the host. This could have a different port number if the original one is no longer available. Before the first data access, you can connect a client of your choice to the network socket. If there is no existing connection when the first data access is made, and the `start_telnet` parameter is true, a host telnet session is started automatically.

The port number of a particular TelnetTerminal instance can be defined when your model system starts. The actual value of the port used by each TelnetTerminal is declared when it starts or restarts, and might not be the value that you specified if the port is already in use. If you are using Model Shell, the port numbers are displayed in the host window in which you started the model.

You can start the TelnetTerminal component in either of the following modes:

Telnet mode
: The terminal component supports a subset of the RFC 854 protocol. This means that the terminal participates in negotiations between the host and client concerning what is and is not supported, but there is no flow control.

Raw mode
: The byte stream passes unmodified between the host and the target. The terminal does not participate in initial capability negotiations between the host and client. Instead it acts as a TCP/IP port. You can use this feature to directly connect to your target through the TelnetTerminal component.

Note

Microsoft Windows 10 disables the Telnet client by default. Follow these steps to enable it:

1. Select **Start** > **Settings**.
2. In the search box, type **Turn Windows features on or off**. The **Windows Features** dialog opens.
3. Select the **Telnet Client** check box and click **OK**. The installation might take several minutes to complete.

Related references

3.7.32 TelnetTerminal on page 3-636

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

1.11 Network set up

This section describes how to set up networks using the serial interface that the TelnetTerminal component exposes.

This section contains the following subsections:

- *1.11.1 User mode networking* on page 1-38.
- *1.11.2 TAP/TUN networking* on page 1-39.

1.11.1 User mode networking

User mode networking emulates a built-in IP router and DHCP server, and routes TCP and UDP traffic between the guest and host. It uses the user mode socket layer of the host to communicate with other hosts.

This allows the use of a significant number of IP network services without requiring administrative privileges, or the installation of a separate driver on the host on which the model is running. Fast Models supports the following kinds of Ethernet device models:

SMSC_91C111
: This is paired with an external HostBridge component. The user mode networking specification is set on the external HostBridge.

VirtioNetMMIO
: This has a built-in HostBridge sub-component. The user mode networking specification is set on the internal HostBridge.

——— **Note** ———

- You can use TCP and UDP over IP, but not ICMP (ping).
- User mode networking does not support forwarding UDP ports on the host to the model.
- You can only use DHCP within the private network.
- You can only make inward connections by mapping TCP ports on the host to the model. This is common to all implementations that provide host connectivity using NAT.
- Operations that require privileged source ports, for example NFS in its default configuration, do not work.
- If setup fails, or the parameter syntax is incorrect, there is no error reporting.

———————————

To enable user mode networking, run the model with the following CADI parameters to activate the components:

SMSC_91C111:

```
-C motherboard.hostbridge.userNetworking=true
-C motherboard.smsc_91c111.enabled=true
```

VirtioNetMMIO:

```
-C motherboard.virtio_net.hostbridge.userNetworking=true
-C motherboard.virtio_net.enabled=true
```

To map a host TCP port to a model port, run the model with the `userNetPorts` parameter. This parameter allows services to appear to be listening on privileged ports in the model but be mapped to unprivileged ports on the host. The syntax is a comma-separated list of items in the form:

```
[host-ip:]hostport=[model-ip:]modelport
```

For example, to map port 8022 on the host to port 22 on the model, use this parameter:

SMSC_91C111:

```
-C motherboard.hostbridge.userNetPorts=8022=22
```


Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VirtioNetMMIO:

```
-C motherboard.virtio_net.hostbridge.userNetPorts=8022=22
```

Either or both of a host IP address and model IP address can optionally be specified on either side of the assignment to select a specific interface on which the mapping will occur. For example:

```
127.0.0.1:8022=127.0.0.1:22
```

The default is to accept connections on any interface.

Related references

3.7.13 HostBridge on page 3-625

1.11.2 TAP/TUN networking

This section describes TAP/TUN networking.

TAP/TUN networking - limitations

This section describes general limitations of TAP/TUN networking for models.

DHCP

If the host uses Dynamic DNS, it inserts records into DNS. If you manage this host with DHCP, then installing TAP networking can cause failure to register in the DNS. After the physical device attaches to the bridge device, the DHCP client reruns, but the DHCP request does not have the correct hostname.

WiFi

Most WiFi adaptors do not implement the required support for TAP networking to work.

Setting up a network connection for Microsoft Windows

This section describes how to set up a network connection.

Prerequisites

The Fast Models networking environment needs the Third Party IP add-on for the Fast Model Portfolio.

Procedure

1. Close all non-essential applications.
2. Install the TAP driver and configure it:
   a. Locate Fast Model Portfolio, default `C:\Program Files\ARM\FastModelPortfolio_X.Y`.
   b. Execute the `ModelNetworking\add_adapter_64.bat` file.
3. Select **Start** > **Control Panel** > **Network Connections** and locate the newly installed TAP device.
4. Select at least one real Ethernet adapter connection. Press the **Ctrl** key to multi-select.
5. Right-click and select **Bridge Connections**.

Configuring the networking environment for Microsoft Windows

This section describes how to set the parameters to make a network connection.

Note

- This procedure has not been tested with wireless network adapters.
- Firewall software might block network traffic in the network bridge, and might result in a networking failure in the model. If the model does not work after configuration, check the firewall settings.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Prerequisites

Use System Canvas or a related Fast Models tool to load a project and build a model, or use Model Shell or Model Debugger to start a prebuilt model.

Procedure

1. Set the appropriate parameters on the `HostBridge` and `SMSC_91C111` components, or on the `VirtioNetMMIO` component and its `HostBridge` subcomponent. For example:
   Example:

   SMSC_91C111:

   ```
   hostbridge.interfaceName="ARM<x>"
   smsc_91c111.enabled=1
   ```

   VirtioNetMMIO:

   ```
   virtio_net.hostbridge.interfaceName="ARM<x>"
   virtio_net.enabled=1
   ```

   `ARM<x>` is an adapter that is built into the network bridge, and is 0 by default. Use `x+1` if `ARM<x>` exists.
2. Optional: Set another parameter on the device, if you have to enable promiscuous mode for the TAP device.
   a. Select **Start** > **Run** > **cmd.exe**.
   b. Enter the following commands.
      Example:

      ```
      netsh bridge show adapter
      netsh bridge set adapter <N> forcecompatmode=enable
      ```

      `<N>` is the id number of the TAP adapter that the first command lists.

Commands for tap_setup_32.exe or tap_setup_64.exe for Microsoft Windows

This section lists the commands for use with `tap_setup_32.exe` or `tap_setup_64.exe`. These files are located in the `ModelNetworking` directory.

`help`
Print help information.

`set_media`
Configure the TAP devices to `Always Connected`.

`set_perm`
Configure the TAP devices to `Non-admin Accessible`.

`restart`
Restart the TAP devices.

`list_dev`
List available TAP devices and output to a file.

`rename`
Rename the device to **ARMx**.

`install <inffile> <id>`
Install a TAP Win32 or Win64 adapter.

`remove <dev>`
Remove TAP Win32 or Win64 adapters. Set ***<dev>*** to `All` to remove all tap devices.

`setup <inffile> <id>`
Automated setup process.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Uninstalling networking for Microsoft Windows

This section describes how to uninstall TAP adapters from the base location of the Fast Models Portfolio, by default `C:\Program Files\ARM\FastModelPortfolio_X.Y`.

Procedure

1. Run the `remove_all_adapters_32.bat` or `remove_all_adapters_64.bat` file. These files are located in the `ModelNetworking` directory.
2. If the uninstallation does not delete the bridge, delete it manually.
   a. Close all non-essential applications.
   b. Select **Network Connections**.
   c. Right-click on the bridge and select **Delete**.

Setting up a network connection for Red Hat Enterprise Linux

This section describes how to set up a network connection.

——— **Note** ———

Perform this procedure once for each host machine.

———————————

——— **Restriction** ———

The set-up and configuration instructions assume that your network provides IP addresses by DHCP. Otherwise, consult your network administrator.

———————————

Prerequisites

Ensure that the `brctl` utility is on your system. This utility is in the installation package, but ARM recommends that you use the standard Linux bridge utilities, which are in the Linux distribution.

Procedure

1. In a shell, change to the `FastModelsPortfolio_X.Y/ModelNetworking` directory.
2. Run the following script from this directory, because it does not work correctly if run from any other location:

   32-bit operating system
   Run `add_adapter_32.sh` as root. For example, `sudo ./add_adapter_32.sh`.

   64-bit operating system
   Run `add_adapter_64.sh` as root. For example, `sudo ./add_adapter_64.sh`.
3. The prompt appears: **Specify the TAP device prefix:(ARM)**. Select **Enter** to accept the default.
4. The prompt appears: **Specify the user list**. Enter a space-separated list of all users who are to use the model on the network, then select **Enter**. All entries in the list must be the names of existing user accounts on the host.
5. The prompt appears: **Enter the network adapter which connects to the network:(eth0)**. Select **Enter** to accept the default, or input the name of a network adapter that connects to your network.
6. The prompt appears: **Enter a name for the network bridge to create:(armbr0)**. Select **Enter** to accept the default, or input a name for the network bridge. You must not have an existing network interface on your system with the selected name.
7. The prompt appears: **Enter the location to write the init script to:(/etc/init.d/FMNetwork)**. Select **Enter** to accept the default, or input another path with a new filename in an existing directory.
8. The prompt appears: **WARNING: the script creates a bridge which includes the local network adapter and tap devices. You may suffer temporary network loss. Do you want to proceed? (Yes or No)**. Verify all values input so far, and enter **Yes**if you want to proceed. If you enter **No**, no changes are made to your system.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

9. A prompt appears to inform you of the changes that the script is to make to your system. Input **Yes** if you are happy to accept these changes, or input **No** to leave your system unchanged.

   ———— **Note** ————

   After entering **Yes**, you might temporarily lose network connectivity. Also, the IP address of the system might change.

   ————————————

Next Steps

The network bridge is disabled after the host system is reset. To configure the host system to support bridged networking, you might have to create links to the `init` script (FMNetwork). The script suggests some appropriate links for Red Hat Enterprise Linux 6.

The default firewall configuration on Red Hat Enterprise Linux 6 blocks packet transmission across the TAP networking bridge device. You can disable the firewall. If the context makes this unwise, then add firewall rules to allow transmission. These `iptables` commands configure the firewall to allow packets across the bridge device:

```
iptables -I FORWARD -m physdev --physdev-is-bridged -j ACCEPT
service iptables save
service iptables restart
```

Setting up a network connection for Ubuntu Linux

This section describes how to set up a network connection.

To use TAP networking with Fast Models on Ubuntu, set up a TAP device manually by following these steps. This guide uses a network interface `eth0` and a username `fmuser`. Replace these values as appropriate.

———— **Note** ————

Typographic errors when modifying network configuration can cause the computer to fail to connect to the network. ARM recommends performing these steps on a machine that you have physical access to.

————————————

Procedure

1. ———— **Note** ————

   Edit the interfaces file with root privileges.

   ————————————

   Add `eth0` to `/etc/network/interfaces` if it is not there.

   This step stops network-manager from managing `eth0`. It can result in network-manager indicating there is no network connection even if there is.

Option	Description

For an interface using DHCP	Add:
	`auto eth0`
	`iface eth0 inet dhcp`

Configure a static IP address	Add the static information, for example:
	`auto eth0`
	`iface eth0 inet static`
	`address 192.168.0.2`
	`netmask 255.255.255.0`
	`gateway 192.168.0.1`

   The network notifier applet launches the GUI tool network-manager. It automatically configures network devices that `/etc/network/interfaces` does not manage, and sets up devices in a way that

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

is incompatible with bridging. This step makes sure that network-manager does not manage the network interface that you want to bridge to. If you are unsure how to configure your network interface, ask your network administrator.

2. Install the bridge-utils package.
   Example: Run:

   ```
   sudo apt-get install bridge-utils
   ```

   Results: This step provides the `brctl` command, for creating and managing the network bridge.

3. Create a bridge device by adding an entry to `/etc/network/interfaces`.
   Example:

   ```
   auto armbr0
   iface armbr0 inet dhcp
   pre-up ifconfig eth0 0.0.0.0 promisc
   post-down ifconfig eth0 0.0.0.0 -promisc
   ```

   The pre-up and post-down lines give commands to execute before bringing up `armbr0` and after bringing it down. These commands put `eth0` into promiscuous mode at pre-up and take it out of promiscuous mode at post-down. Promiscuous mode makes sure that the hardware does not filter out packets for the virtual ethernet device.

   Results: This step creates a bridge device that is called `armbr0` from Fast Models TAP devices to the physical network.

4. Create the TAP devices.

   TAP devices need permission for specific users, so create one for each user who is to run the model with the virtual ethernet device. For example, to create a TAP device that is called `ARMfmuser` for the user `fmuser`, add the following lines to the `armbr0` section of `/etc/network/interfaces`.

   Example:

   ```
   pre-up ip tuntap add dev ARMfmuser mode tap user fmuser
   pre-up ifconfig ARMfmuser 0.0.0.0 promisc
   post-down ip tuntap del dev ARMfmuser mode tap
   ```

   Results: This step creates a TAP device for each user.

5. Create a bridge between the TAP devices and the network interface `eth0` by adding a `bridge_ports` line to the `armbr0` section of `/etc/network/interfaces`. For example, for a TAP device that is named `ARMfmuser`, add the following line:
   Example:

   ```
   bridge_ports eth0 ARMfmuser
   ```

6. The added `/etc/network/interfaces` code now looks like the following:
   Example:

   ```
   auto armbr0
   iface armbr0 inet dhcp
   pre-up ifconfig eth0 0.0.0.0 promisc
   post-down ifconfig eth0 0.0.0.0 -promisc
   pre-up ip tuntap add dev ARMfmuser mode tap user fmuser
   pre-up ifconfig ARMfmuser 0.0.0.0 promisc
   post-down ip tuntap del dev ARMfmuser mode tap
   bridge_ports eth0 ARMfmuser
   ```

7. Restart network services with one of the following options:
   - Run the following commands:

     ```
     sudo ifdown eth0 && sudo ifup eth0
     sudo ifup armbr0
     sudo service network-manager restart
     ```

     Note

     `armbr0` must be explicitly started.

   - Restart the computer.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Results: Both choices disconnect and reconnect all network interfaces.

Configuring the networking environment for Linux

This section describes how to set the parameters to make a network connection.

Note

Firewall software might block network traffic in the network bridge, and result in a networking failure. If the model does not work after configuration, check the firewall settings.

Prerequisites

Use System Canvas or a related Fast Models tool to load a project or model, and then select a component.

Procedure

1. Set the parameters on the `HostBridge` and `SMSC_91C111` components, or on the `VirtioNetMMIO` component and its `HostBridge` subcomponent. For example:
   Example:

   SMSC_91C111:

   ```
   hostbridge.interfaceName=ARM<username>smsc_91c111.enabled=1
   ```

   VirtioNetMMIO:

   ```
   virtio_net.hostbridge.interfaceName=ARM<username>virtio_net.enabled=1
   ```

   `ARM<username>` is an adapter that is built into the network bridge.

Related information

Fast Models User Guide

Solutions to networking issues on Linux

This section describes how to solve networking issues.

The model networking works after initial setup, but stops working after reboot

Set the correct access permissions for the `/dev/net/tun` device, by executing `chmod 666 /dev/net/tun` as root. To preserve the change across reboots, modify the udev rules of the TAP device: open `/etc/udev/rules.d/50-udev.rules` as root, and find the line:

```
KERNEL=="tun", NAME="net/%k"
```

If it does not have `MODE="0666"` at the end of the line, append `MODE="0666"`:

```
KERNEL=="tun", NAME="net/%k", MODE="0666"
```

Model networking installs correctly, but when a model starts up, the model cannot receive packets

Disable the firewall on the host machine, or add the TAP device to `trusted devices`.

Note

Refer to the vendor documentation manual.

Disabling and re-enabling networking for Linux

This section describes how to disable and re-enable networking with an `init` script.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Caution

These operations remove/restore TAP devices and the network bridge. There is a temporary loss of network connectivity and your IP address might change.

Procedure

- To disable networking without uninstalling it, invoke the installed `init` script (by default, `/etc/init.d/FMNetwork`) as root with the parameter `stop`.
  Example:

```
sudo /etc/init.d/FMNetwork stop
```

- To re-enable networking, invoke the `init` script as root with the parameter `start`.
  Example:

```
sudo /etc/init.d/FMNetwork start
```

Uninstalling networking for Linux

This section describes the steps to uninstall a network.

Procedure

1. In a shell, change to the `/FastModelsPortfolio_X.Y/ModelNetworking` directory.
2. Run `uninstall.sh` as root, passing the location of the `init` script (FMNetwork):
   Example: `sudo ./uninstall.sh /etc/init.d/FMNetwork`

   You must run this script from the directory in which it is installed, because it does not work correctly if run from any other location.

   Caution

   There is a temporary loss of network connectivity and your IP address might change.

Next Steps

The uninstall script removes everything that can be safely removed. It does not remove:

- symlinks to the `init` script.
- `/sbin/brctl`.

You must remove any symlinks that you have created. Removing `brctl` is optional.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

1.12 Model Trace Interface

This section describes the *Model Trace Interface* (MTI).

Fast Models supports the generation of traces that consistently track the execution and related activities in the model. In particular, tracking those activities that affect the state of the modeled IP. Generated virtual platforms provide trace support by using plug-ins in the form of DLLs and shared objects on Windows and Linux, respectively.

The following diagram represents the MTI architecture, where all the tracing information of the system is found.

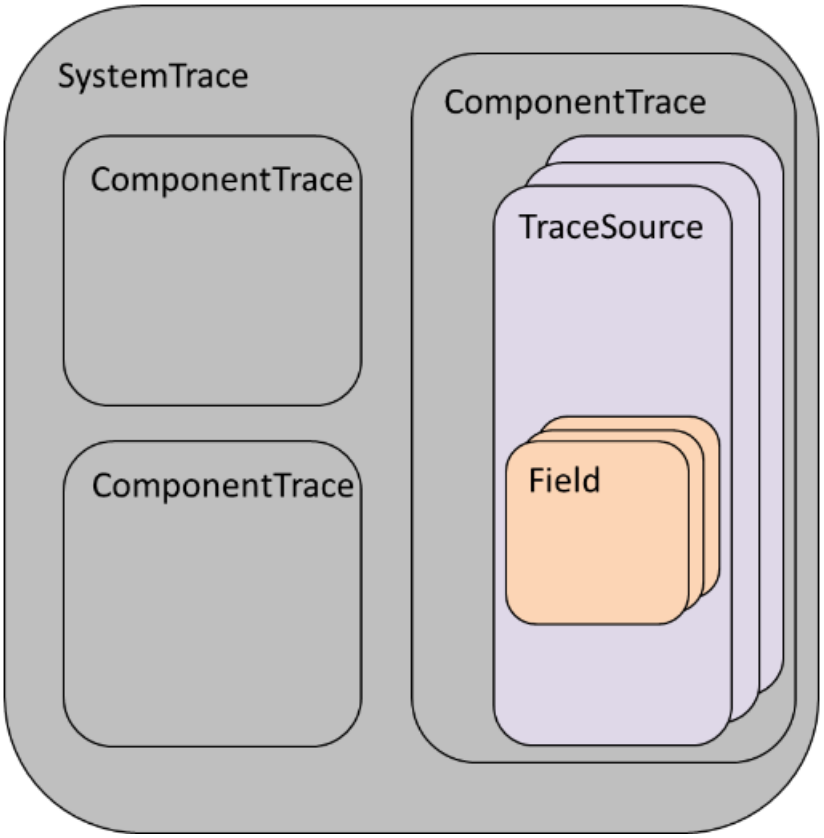

Figure 1-2 MTI architecture

A trace source provides information about a specific event that occurs in the component. In a processor component, this event can be, for example, the execution of an instruction, the taking of a branch, or an MMU translation. It can also be an event that is specific to models, such as the `SYNC` event, which is called at every quantum boundary. Each trace source contains fields that give more information about the event, including a text description. When the field is of type `MTI_ENUM`, the values for this field are also listed. For example:

```
Source CACHE_READ_MISS (Read access cache miss.)
    Field IS_SHARED type:MTI_ENUM size:1 (Is the access shared)
        0x0 = NON_SHARED
        0x1 = SHARED
    Field IS_PRELOAD type:MTI_ENUM size:1 (Is the access a preload)
        0x0 = NOT_PRELOAD
        0x1 = PRELOAD
    Field MASTER_ID type:MTI_UNSIGNED_INT size:4 (Indicates the master associated with this
monitor)
```

After you enable or disable a trace source, there might be a delay before the change takes effect, depending on the current syncLevel. See *1.8.1 syncLevel definitions* on page 1-29 for more information.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Arm produces several pre-built trace plug-ins, which are documented in *Plug-ins for Fast Models* on page 5-2890. Source code for these plug-ins and for other example plug-ins is provided in `$PVLIB_HOME/examples/MTI/`.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

1.13 Using parameters to set port values

Some processor and peripheral component ports are almost always static in value when used as part of a typical platform. For example, the reset vector base address register address (RVBARADDR) port in processor components.

To facilitate easy configuration of platform models, the IP models for these components can provide a shadow parameter for these ports. This parameter can be used to change the value that is used by the model. In these cases, the following rules apply:

- If a port is driven in the platform model, then the parameter value is ignored.
- If a port is not driven in the platform model, then the parameter value is sampled at both simulator reset, and at every subsequent simulation reset of the specific IP model.

  Note

  Simulator reset corresponds with the LISA `reset()` behavior and the SystemC `start_of_simulation()` callback.

- All ports and parameters that are sampled at reset are sampled when the simulation reset signal concerned is deasserted.
- If a port is not driven in the platform model, and a parameter has not been set, then the default value for the parameter is used.

In some IP models, the value of some ports can only be set by using a parameter. That is, the parameter is provided instead of the port.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

1.14 About PVBus system components

A bus system consists of a number of bus masters, some infrastructure and a number of bus slaves.

Each bus master must contain a `PVBusMaster` subcomponent, and each bus slave must contain a `PVBusSlave` subcomponent. These subcomponents provide `PVBus` master and slave ports. Each `PVBus` master port can only connect to one slave, but any number of other masters can connect to the same slave. `PVBusDecoder`, `PVBusMaster` and `PVBusSlave` components communicate using the `PVBus` protocol.

`PVBusDecoder` components can be added to the bus system. Each of these permits its masters to connect to multiple slaves, each associated with a different bus address range.

`PVBusSlave` subcomponents provide built-in support for declaring memory-like, device-like, abort or ignore address ranges. `PVBus` has support for dealing efficiently with memory-like devices such as RAM, ROM, and Flash.

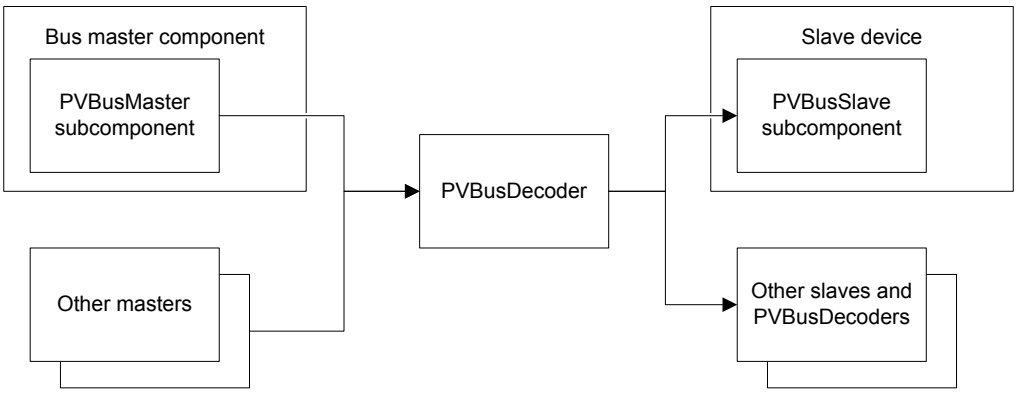

Figure 1-3 Sample bus topology

All communication over the `PVBus` is performed using transactions that are generated by `PVBusMaster` subcomponents and fulfilled by `PVBusSlave` components. Transactions have a 32-bit Master ID, which is the ID of the bus master. Transactions can be routed to the slave device through its `PVBusSlave` subcomponent. When configured, the `PVBusSlave` can handle memory-like transactions efficiently without having to route these transactions to the slave device. Transactions are atomic unless slave devices block transactions, for example an SMMU with stall mode enabled. A slave device that can block transactions and all its upstream bus components must be re-entrant safe for bus transactions.

Fast Models provides some example `PVBus` systems:

- `$PVLIB_HOME/examples/LISA/common/LISA/RemapDecoder.lisa`. This example dynamically modifies routing of requests based on a remap signal, using the `TZSwitch` component.
- The directory `$PVLIB_HOME/examples/LISA/BusComponents/` contains a set of example components that show various ways of using the `PVBus` interface.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

1.15 PVBus C++ transaction and Tx_Result classes

This section describes the C++ transaction and `Tx_Result` classes.

This section contains the following subsections:

- *1.15.1 Class pv::TransactionGenerator* on page 1-50.
- *1.15.2 TransactionGenerator efficiency considerations* on page 1-50.
- *1.15.3 Enum pv::AccessWidth* on page 1-51.
- *1.15.4 Class pv::Transaction* on page 1-51.
- *1.15.5 Class pv::ReadTransaction* on page 1-51.
- *1.15.6 Class pv::WriteTransaction* on page 1-52.

1.15.1 Class pv::TransactionGenerator

This class provides efficient mechanisms for bus masters to generate transactions that are transmitted over the `pvbus_m` port of the associated PVBusMaster subcomponent.

You can produce pv::TransactionGenerator objects by invoking the `createTransactionGenerator()` method on the control port of a PVBusMaster component.

```
class pv::TransactionGenerator
{
    // Tidy up when TransactionGenerator is deleted.
    ~TransactionGenerator()

    // Control AXI-specific signal generation for future transactions.
    // Privileged processing mode.
    void setPrivileged(bool priv = true);

    // Instruction access (vs data).
    void setInstruction(bool instr = true);

    // Normal-world access (vs secure).
    void setNonSecure(bool ns = true);

    // Locked atomic access.
    void setLocked(bool locked = true);

    // Exclusive atomic access.
    void setExclusive(bool excl = true);

    // Generate transactions.
    // Generate a read transaction.
    bool read(bus_addr_t, pv::AccessWidth width, uint32_t *data);

    // Generate a write transaction.
    bool write(bus_addr_t, pv::AccessWidth width, uint32_t const *data);

    // Generate read transactions.
    bool read8(bus_addr_t, uint8_t *data);
    bool read16(bus_addr_t, uint16_t *data);
    bool read32(bus_addr_t, uint32_t *data);
    bool read64(bus_addr_t, uint64_t *data);

    // Generate write transactions.
    bool write8(bus_addr_t, uint8_t const *data);
    bool write16(bus_addr_t, uint16_t const *data);
    bool write32(bus_addr_t, uint32_t const *data);
    bool write64(bus_addr_t, uint64_t const *data);
};
```

1.15.2 TransactionGenerator efficiency considerations

TransactionGenerators are most efficient for multiple accesses to one 4KB page.

Each TransactionGenerator caches connection information internally. This improves efficiency for multiple accesses to a single 4KB page. If a component requires repeated access data from different pages, for example when streaming from one location to another, ARM recommends you create a TransactionGenerator for each location.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

You can dynamically create and destroy TransactionGenerators, but it is better to allocate them once at initialization and destroy them at shutdown. See the DMA example in the examples directory of the System Canvas Model Library distribution.

1.15.3 Enum pv::AccessWidth

This enum selects the required bus width for a transaction.

Defined values are:

- `pv::ACCESS_8_BITS`
- `pv::ACCESS_16_BITS`
- `pv::ACCESS_32_BITS`
- `pv::ACCESS_64_BITS`

1.15.4 Class pv::Transaction

This class is a base class for read and write transactions that are visible in the PVBusSlave subcomponent. It contains functionality common to both types of transaction.

This class provides an interface that permits bus slaves to access the details of the transaction. Do not instantiate these classes manually. The classes are generated internally by the PVBus infrastructure.

This base class provides access methods to get the transaction address, access width, and bus signals. It also provides a method to signal that the transaction has been aborted.

```
class pv::Transaction
{
    public:
    // Accessors
    bus_addr_t getAddress() const;          // Transaction address.
    pv::AccessWidth getAccessWidth() const; // Request width.
    int getAccessByteWidth() const;         // Request width in bytes.
    int getAccessBitWidth() const;          // Request width in bits.

    bool isPrivileged() const;  // Privileged process mode?
    bool isInstruction() const; // Instruction request vs data?
    bool isNonSecure() const;   // Normal-world vs secure-world?
    bool isLocked() const;      // Atomic locked access?
    bool isExclusive() const;   // Atomic exclusive access?
    uint32_t getMasterID() const;
    bool hasSideEffect() const;

    // Generate transaction returns
    Tx_Result  generateAbort();          // Cause the transaction to abort.
    Tx_Result  generateSlaveAbort();     // Cause the transaction to abort.
    Tx_Result  generateDecodeAbort();    // Cause the transaction to abort.
    Tx_Result  generateExclusiveAbort(); // Cause the transaction to abort.
    Tx_Result  generateIgnore();
};
```

1.15.5 Class pv::ReadTransaction

This class extends the pv::Transaction class to provide methods for returning data from a bus read request.

```
class ReadTransaction : public Transaction
{
    public:
    /*! Return a 64-bit value on the bus. */
    Tx_Result setReturnData64(uint64_t);

    /*! Return a 32-bit value on the bus. */
    Tx_Result setReturnData32(uint32_t);

    /*! Return a 16-bit value on the bus. */
    Tx_Result setReturnData16(uint16_t);

    /*! Return an 8-bit value on the bus. */
    Tx_Result setReturnData8(uint8_t);

    /*! This method provides an alternative way of returning a Tx_Result
    * success value (instead of just using the value returned from
    * setReturnData<n>()).
    *
```


Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

```
    * This method can only be called if one of the setReturnData<n>
    * methods has already been called for the current transaction.
    */
    Tx_Result readComplete();
};
```

1.15.6 Class pv::WriteTransaction

This class extends the pv::Transaction class to provide methods for returning data from a bus write request.

```
class WriteTransaction : public Transaction
{
    public:
    /*! Get bottom 64-bits of data from the bus. If the transaction width
    * is less than 64-bits, the data will be extended as appropriate.
    */
    uint64_t getData64() const;

    /*! Get bottom 32-bits of data from the bus. If the transaction width
    * is less than 32-bits, the data will be extended as appropriate.
    */
    uint32_t getData32() const;

    /*! Get bottom 16-bits of data from the bus. If the transaction width
    * is less than 16-bits, the data will be extended as appropriate.
    */
    uint16_t getData16() const;

    /*! Get bottom 8-bits of data from the bus. If the transaction width
    * is less than 8-bits, the data will be extended as appropriate.
    */
    uint8_t getData8() const;

    /*! Signal that the slave has handled the write successfully.
    */
    Tx_Result writeComplete();
};
```


Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

1.16 Visualisation Library

This section describes the components and protocols included in the Visualisation Library for Fast Models.

This section contains the following subsections:

- *1.16.1 Visualisation Library - about* on page 1-53.
- *1.16.2 LISA Visualisation models* on page 1-53.
- *1.16.3 Visualisation Library C++ classes* on page 1-54.

1.16.1 Visualisation Library - about

The Visualisation Library does not model hardware directly but instead provides components, protocols, and a library. These permit a GUI display that lets you interact with the external I/O from the model platform.

The types of I/O handled include:

- LCD display, such as the output from the PL110_CLCD component display port.
- LEDs representing values from a ValueState port as either single lights, or as segmented alphanumeric displays.
- DIP switches, which can drive a ValueState port.
- Capture of keyboard and mouse input, using the KeyboardStatus and MouseStatus protocols to feed input to a PS2Keyboard or PS2Mouse component.
- Background graphics, custom rendered graphics, and clickable push buttons, permitting the UI to provide display a skin representing the physical appearance of the device being modeled.
- Status information such as processor instruction counters, with values taken from the InstructionCount port of a processor.

The Visualisation Library provides a C++ API that enables you to write your own visualization components in LISA+. These custom components can display any combination of the supported I/O types.

You can add the prebuilt GUIPoll component to your custom component. The GUIPoll component provides a LISA visualization component with a periodic signal that keeps the display updated, even when the simulation is stopped.

The Visualisation Library supports one signaling protocol, the LCD protocol.

Related references

2.4.7 LCD protocol on page 2-72

1.16.2 LISA Visualisation models on page 1-53

1.16.3 Visualisation Library C++ classes on page 1-54

1.16.2 LISA Visualisation models

The Visualisation components provide a host window to display status information in addition to a frame buffer.

Each example platform model, such as the VE, contains its own LISA Visualisation model. You can use the model as the basis for your own visualization containing components, such as a PL110_CLCD component. To use the Visualisation components in your own system, copy the component LISA files from the relevant platform model directory, because they are not in the generic Model Library.

Related references

Chapter 9 Versatile Express Model on page 9-2999

Related information

Fast Models User Guide

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

1.16.3 Visualisation Library C++ classes

This section describes the Visualisation Library C++ classes and structures.

C++ classes inclusion

To use these Visualisation Library classes, begin your LISA component with the correct `#include` statement.

```
includes
{
#include “components/Visualisation.h”
}
```

Visualisation class

This class provides access to the API for creating a custom LISA visualization component.

Visualisation class - about

A component obtains an instance of this class by calling the global function `createVisualisation()`. The component can then use this instance to control the size and layout of the visualization window.

The Visualisation Library is implemented using the *Simple DirectMedia Layer* (SDL) cross-platform rendering library.

Global functions

The global function is `createVisualisation()`.

`Visualisation *createVisualisation()`
This generates an instance of the Visualisation Library. You can only call this function once, because SDL only supports opening a single display window.

Class Visualisation

This section describes the methods of the Visualisation class.

`~Visualisation()`
Destructor for the Visualisation Library. You must only call this method when your simulation is shutting down, after all allocated resources (VisRenderRegions, VisPushButtonRegions, VisBitmaps) have been deleted.

`void configureWindow(unsigned int width, unsigned int height, unsigned int bit_depth)`
This sets the visualization window to the requested size and bit depth. Depending on the display capabilities, the window might actually get a different bit depth from the size you requested.

`VisBitmap *loadImage(char const *filename)`
This allocates a new VisBitmap object, initialized by loading a Microsoft Windows Bitmap (`.BMP`) from the given file.

`VisBitmap *loadImageWithAlphaKey(char const *filename, unsigned int red, unsigned int green, unsigned int blue)`
This allocates a VisBitmap object, as with `loadImage()`. All pixels of the color specified by `red`, `green`, `blue` are converted into a transparent alpha channel.

`VisBitmap *cropImage(VisBitmap *source, int x, int y, unsigned int width, unsigned int height)`
This allocates a new VisBitmap object, by cropping a region from the source bitmap.

`void releaseImage(VisBitmap *)`
This releases the resources held by the given VisBitmap. The underlying bitmap is only be unloaded if it is not in use.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

`void setBackground(VisBitmap *background, int x, int y)`
This sets the background image for the visualization window. This takes a copy of the data referenced by the VisBitmap, so it is safe for the client to call `releaseImage(background)` immediately after calling `setBackground()`. The background is not displayed until the first call to `poll()`.

`VisRenderRegion *createRenderRegion()`
This allocates a new VisRenderRegion object that can be used to display arbitrary graphics, including LCD contents, in a rectangular region.

`VisPushButtonRegion *createPushButtonRegion()`
This allocates a new VisPushButtonRegion, which can be placed at a location on the display to provide a clickable push button.

`bool poll(VisEvent *event)`

This call permits the Visualisation Library to poll for GUI events. The client passes a reference to a VisEvent structure, which receives details of a single mouse/keyboard event.

The method returns false if no events have occurred.

Your LISA visualization implementations must call this periodically by using a GUIPoll component. On each `gui_callback()` event, you must ensure that the visualization component repeatedly calls `poll()` until it returns false.

`void lockMouse(VisRegion *region)`
This method locks the mouse to the visualization window and hides the mouse pointer.

`void unlockMouse()`
This unlocks and redisplays the mouse pointer.

`bool hasQuit()`
This returns true if the user has clicked on the close icon of the visualization window.

Class VisRegion

This class is the common base class for VisPushButtonRegion and VisRenderRegion, representing a region of the visualization display.

`~VisRegion()`
Permits clients to delete a VisPushButtonRegion when it is no longer required.

`void setId(void *id)`
Permits a client-defined identifier to be associated with the region.

`void *getId()`
Returns the client-defined identifier.

`void setVisible(bool vis)`
Specifies whether the region is to be displayed on the screen. This is currently ignored by the SDL implementation.

`void setLocation(int x, int y, unsigned int width, unsigned int height)`
Sets the location of this region relative to the visualization window.

Class VisPushButtonRegion : public VisRegion

This class defines a region of the visualization window that represents a clickable button.

Optionally, the button can provide different VisBitmap representations for a button-up and a button-down graphic, and a graphic to use when the mouse pointer rolls over the button.

In addition to the public method defined in VisRegion, this class defines these methods:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

`void setButtonUpImage(VisBitmap *bmpUp) : void`

`setButtonDownImage(VisBitmap *bmpDown) : void`

`setButtonRollOverImage(VisBitmap *bmpRollover)`

Sets the graphics to be used for each of the button states. If any image is not specified or is set to NULL, then the corresponding area of the visualization background image is used.

The VisPushButtonRegion takes a copy of the VisBitmap, so the client can safely call Visualisation::releaseBitmap() on its copy.

`void setKeyCode(int code)`
This sets the code for the keypress event that is generated when the button is pressed or released.

Class VisRenderRegion : public VisRegion

This class defines a region of the visualization window that can render client-drawn graphics, including a representation of the contents of an LCD.

In addition to the public method defined in VisRegion, the class defines these methods:

`VisRasterLayout const *lock()`
Locks the region for client rendering. While the buffer is locked, the client can modify the pixel data for the buffer. You must not call the methods `writeText()` and `renderBitmap()` while the buffer is locked.

`void unlock()`
Releases the lock on the render buffer, permitting the buffer to be updated on screen.

`void update(int left, int top, unsigned int width, unsigned int height)`
Causes the specified rectangle to be drawn to the GUI.

`int writeText(const char *text, int x, int y)`
Renders the given ASCII text onto an unlocked VisRenderRegion. The return value is the x co-ordinate of the end of the string. The default font is 8 pixels high, and cannot be changed.

void `renderBitmap(VisBitmap *bitmap, int x, int y)`
Draws a bitmap onto an unlocked VisRenderRegion.

Struct VisRasterLayout

This struct defines the layout of the pixel data in a frame-buffer.

The `lock()` method of the LCD protocol expects to be given a pointer to this structure. You can generate a suitable instance by calling `VisRasterRegion::lock()`.

The structure contains these fields:

`uint8_t *buffer`
This points to the buffer for the rasterized pixel data. The controller can write pixels into this buffer, but must stick within the bounds specified by the width and height.

`uint32_t pitch`
The number of bytes between consecutive raster lines in the pixel data. This can be greater than the number of bytes per line.

`uint32_t width`
The width, in pixels, of the render area. This value can be less than the width requested by the LCD controller when it called `lock()`.

`uint32_t height`
The height, in pixels, of the render area. This value can be less than the height requested by the LCD controller when it called `lock()`.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

`VisPixelFormat` format
: This structure defines the format of the pixel data in the raster buffer.

`bool` changed
: This is set to true if the pixel format or buffer size has changed since the previous call to `lock()`.

Pixel data is represented as a one-dimensional array of bytes. The top-left pixel is pointed to by the buffer member. Each pixel takes up a number of bytes, given by `format.pbytes`.

The pixel at location (x, y) is stored in the memory bytes starting at `buffer[y * pitch + x * format.pbytes]`.

Struct VisPixelFormat

This struct specifies the format of pixel data within the buffer.

The members are:

`uint32_t rbits, gbits, bbits`
: The number of bits per color channel.

uint32_t `roff, goff, boff`
: The offset within the pixel data value for the red/green/blue channels.

uint32_t `pbytes`
: The size of a single pixel, in bytes.

`format.pbytes` specifies the number of bytes that make up the data for a single pixel. These bytes represent a single pixel value, stored in host-endian order. The pixel value contains a number of the form `(R<<format.roff) + (G<<format.goff) + (B<<format.boff)`, where `(R,G,B)` represents the values of the color channels for the pixel, containing values from 0 up to `(1<<format.rbits)`, `(1<<format.gbits)`, `(1<<format.bbits)`.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Chapter 2
Protocols

Components communicate through connected ports. Ports have protocols that define the function calls for different connections.

It contains the following sections:

- *2.1 AMBA-PV protocols* on page 2-59.
- *2.2 Clocking protocols* on page 2-65.
- *2.3 Debug interface protocols* on page 2-67.
- *2.4 Peripheral protocols* on page 2-71.
- *2.5 Power management protocols* on page 2-80.
- *2.6 Processor protocols* on page 2-81.
- *2.7 Signaling protocols* on page 2-85.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

2.1 AMBA-PV protocols

This section describes the AMBA-PV protocols.

This section contains the following subsections:

- *2.1.1 AMBA-PV protocols - about* on page 2-59.
- *2.1.2 AMBAPV protocol* on page 2-59.
- *2.1.3 AMBAPVACE protocol* on page 2-61.
- *2.1.4 AMBAPVSignal protocol* on page 2-63.
- *2.1.5 AMBAPVSignalState protocol* on page 2-63.
- *2.1.6 AMBAPVValue protocol* on page 2-63.
- *2.1.7 AMBAPVValue64 protocol* on page 2-63.
- *2.1.8 AMBAPVValueState protocol* on page 2-64.
- *2.1.9 AMBAPVValueState64 protocol* on page 2-64.

2.1.1 AMBA-PV protocols - about

The AMBA-PV components and protocols permit you to model a platform that interfaces with an ARM® AMBA-based system.

The protocols are:

AMBAPV
Models the AMBA protocols AXI4, AXI3, AHB and APB.

AMBAPVACE
Models the AMBA ACE and DVM protocols.

AMBAPVSignal
Models interrupts.

AMBAPVSignalState
Transfers and receives signal states.

AMBAPVValue
Models propagation of 32-bit integer values between components.

AMBAPVValue64
Models propagation of 64-bit integer values between components.

AMBAPVValueState
Permits a master to retrieve the current value from a slave, using 32-bit integer values.

AMBAPVValueState64
Permits a master to retrieve the current value from a slave, using 64-bit integer values.

There are ready-to-use components that provide you with conversions between protocols.

Related references

3.1 Bridge components on page 3-88

Related information

AMBA-PV Extensions to TLM 2.0 Developer Guide

2.1.2 AMBAPV protocol

The AMBAPV protocol defines behaviors for single read and single write transactions. This covers ARM AMBA AXI4, AXI3, AHB and APB bus protocol families, at the PV level.

In addition, the AMBAPV protocol provides support for AMBA protocol additional control information:

- Protection units.
- Exclusive access and locked access mechanisms.
- System-level caches.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

The behaviors of the protocol are:

read()

Completes a single read transaction at the given address for the given size in bytes. Additional AMBA protocol control information can be specified using the `ctrl` parameter. The `socket_id` parameter must be set to 0 in this context.

```
read(int socket_id, const sc_dt::uint64 &addr,
    unsigned char *data, unsigned int size,
    const amba_pv::amba_pv_control *ctrl,
    sc_core::sc_time &t):
    amba_pv::amba_pv_resp_t
```

write()

Completes a single write transaction at the given address with specified data and write strobes. The size of the data is specified in bytes. Additional AMBA protocol control information can be specified using the `ctrl` parameter. The `socket_id` parameter must be set to 0 in this context.

```
write(int socket_id, const sc_dt::uint64 &addr,
    unsigned char *data, unsigned int size,
    const amba_pv::amba_pv_control *ctrl,
    unsigned char *strb, sc_core::sc_time &t):
    amba_pv::amba_pv_resp_t
```

debug_read()

Completes a debug read transaction from a given address without causing any side effects. Specify the number of bytes to read in the `length` parameter. The number of successfully read values is returned. Additional AMBA protocol control information can be specified in the `ctrl` parameter. The `socket_id` parameter must be set to 0 in this context. This behavior is empty by default and returns 0.

```
debug_read(int socket_id, const sc_dt::uint64 &addr,
    unsigned char *data, unsigned int length,
    const amba_pv::amba_pv_control *ctrl):
    unsigned int
```

debug_write()

Completes a debug write transaction to a given address without causing any side effects. Specify the number of bytes to write in the `length` parameter. The number of successfully written values is returned. Additional AMBA protocol control information can be specified in the `ctrl` parameter. The `socket_id` parameter must be set to 0 in this context. This behavior is empty by default and returns 0.

```
debug_write(int socket_id, const sc_dt::uint64 &addr,
    unsigned char *data, unsigned int length,
    const amba_pv::amba_pv_control *ctrl):
    unsigned int
```

b_transport()

This is an optional slave behavior for blocking transport. It completes a single transaction using the blocking transport interface. The `amba_pv::amba_pv_extension` must be added to the transaction before calling this behavior. The `socket_id` parameter must be set to 0 in this context.

```
b_transport(int socket_id,
    amba_pv::amba_pv_transaction &trans, sc_core::sctime &t)
```


Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

transport_dbg()

This optional slave behavior implements the TLM debug transport interface. An `amba_pv::amba_pv_extension` object must be added to the transaction before calling this behavior. The `socket_id` parameter must be set to 0 in this context.

```
transport_dbg(int socket_id, amba_pv::amba_pv_transaction &trans, sc_core::sctime
&t): unsigned int
```

get_direct_mem_ptr()

This is an optional slave behavior for requesting a DMI access to a given address. It returns a reference to a DMI descriptor that contains the bounds of the granted DMI region. Returns `true` if a DMI region is granted, `false` otherwise.

The `amba_pv::amba_pv_extension` must be added to the transaction before calling this behavior. The `socket_id` parameter must be set to 0 in this context.

```
get_direct_mem_ptr(int socket_id,
    amba_pv::amba_pv_transaction &trans,
    tlm::tlm_dmi &dmi_data):
    bool
```

invalidate_direct_mem_ptr()

This is an optional master behavior to invalidate a DMI request. It invalidates DMI pointers that were previously established for the given DMI region. The `socket_id` parameter is 0 in this context.

```
invalidate_direct_mem_ptr(int socket_id,
    sc_dt::uint64 start_range, sc_dt::uint64 end_range)
```

The generic payload data is formatted as an array of bytes in order of ascending bus address. This means that irrespective of the host machine endianness or modeled bus width, a little endian master must write the bytes of a word in increasing significance as the array index increases and a big endian master must write the bytes of a word in decreasing significance as the array index increases. A master or slave whose endianness does not match the endianness of the host machine must endian swap any access to the payload data that is wider than one byte. The same byte ordering rule applies to memory accesses using DMI pointers.

2.1.3 AMBAPVACE protocol

This protocol defines behaviors for bus transactions. This covers ARM AMBA ACE and DVM bus protocol families, all at the PV level.

In addition, this protocol provides support for AMBA protocol additional extension information:

- Secure and privileged accesses.
- Atomic operations.
- System-level caching and buffering control.
- Cache coherency transactions (ACE-Lite).
- Bi-directional cache coherency transactions (ACE).
- Distributed virtual memory transactions (DVM).

The behaviors of the protocol are:

b_transport()

This slave behavior implements the TLM blocking transport interface. An `amba_pv::amba_pv_extension` object must be added to the transaction before calling this behavior. The `socket_id` parameter must be set to 0 in this context.

```
b_transport(int socket_id,  amba_pv::amba_pv_transaction &trans, sc_core::sctime &t)
```


Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

transport_dbg()

This optional slave behavior implements the TLM debug transport interface. You must add an `amba_pv::amba_pv_extension` object to the transaction before calling this behavior, setting the `socket_id` parameter to 0 in this context.

```
transport_dbg(int socket_id,  amba_pv::amba_pv_transaction &trans, sc_core::sctime
&t):  unsigned int
```

get_direct_mem_ptr()

This optional slave behavior is for requesting a DMI access to a given address. It returns a reference to a DMI descriptor that contains the bounds of the granted DMI region. Returns true if a DMI region is granted, false otherwise. You must add an `amba_pv::amba_pv_extension` object to the transaction before calling this behavior, setting the `socket_id` parameter to 0 in this context.

```
get_direct_mem_ptr(int socket_id,  amba_pv::amba_pv_transaction &trans,
tlm::tlm_dmi &dmi_data):  bool
```

b_snoop()

This master behavior implements an upstream snooping TLM blocking transport interface. You must add an `amba_pv::amba_pv_extension` object to the transaction before calling this behavior, setting the `socket_id` parameter to 0 in this context.

```
b_snoop(int socket_id,  amba_pv::amba_pv_transaction &trans, sc_core::sctime &t)
```

snoop_dbg()

This optional master behavior implements an upstream snooping TLM debug transport interface. You must add an `amba_pv::amba_pv_extension` object to the transaction before calling this behavior, setting the `socket_id` parameter to 0 in this context.

```
snoop_dbg(int socket_id,  amba_pv::amba_pv_transaction &trans, sc_core::sctime &t):
unsigned int
```

invalidate_direct_mem_ptr()

Use this optional master behavior to invalidate a DMI request. It invalidates DMI pointers that were previously established for the given DMI region. The `socket_id` parameter is 0 in this context.

```
invalidate_direct_mem_ptr(int socket_id,  sc_dt::uint64 start_range, sc_dt::uint64
end_range)
```

The generic payload data is in the format of an array of bytes in order of ascending bus address. This means that irrespective of the host machine endianness or modeled bus width, a little endian master must write the bytes of a word in increasing significance as the array index increases and a big endian master must write the bytes of a word in decreasing significance as the array index increases. A master or slave whose endianness does not match the endianness of the host machine must endian swap any access to the payload data that is wider than one byte. The same byte ordering rule applies to memory accesses using DMI pointers.

Special considerations for ACE and cache coherent interconnects

To maintain memory coherency, apply these rules for debug transactions.

An ACE interconnect model must be able to cope with concurrent transactions in accordance with the hazard avoidance and prioritization rules in the ACE specification. Any external bus request, downstream transaction or upstream snoop transaction, can potentially cause a transaction to stall and the calling thread to be blocked, resulting in any number of other threads being scheduled.

To maintain memory coherency, apply these rules for debug transactions:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

debug reads
The bus must return data that represents the values that the bus master expects to observe if it issues a bus read. This must not modify the state of any bus components.

debug writes
must modify the contents of all copies of the location being accessed, so that a subsequent read from this location returns the data in the debug-write request. The debug write must not modify any other state, for example such as cache tags, clean/dirty/shared/unique MOESI state.

The implications for a coherent interconnect are that incoming debug transactions must be broadcast back upstream as debug snoop transactions to all ports other than the one the request came in on. Incoming debug snoops must propagate upwards. Debug reads can terminate as soon as they hit a cache. Debug writes must continue until they propagate to all possible copies of the location, including downstream to main memory.

For cases where a debug transaction hazards with non-debug transactions that are in-flight, the debug transaction must observe a weak memory-order model. Any component that can block a thread whilst responsible for the payload of an in-flight transaction must take particular care. In these cases, the debug transaction must be hazarded against the in-flight payload to ensure that debug reads do not return stale data and debug writes do not cause cache incoherency.

Only use DMI when you can guarantee that subsequent transactions do not result in any state transitions. This means, in general, do not use DMI for ACE coherent cacheable transactions.

2.1.4 AMBAPVSignal protocol

This protocol defines a single behavior to permit masters to change the state of signals such as interrupt. Strictly speaking, AMBA3 does not cover this behavior, but the AMBA-PV components do provide it.

`set_state(int export_id, const bool &state): void`
Transfers a signal state. The `export_id` parameter must be set to 0 in this context.

2.1.5 AMBAPVSignalState protocol

This protocol defines two behaviors that permit a master to change the state of signals such as interrupt and to retrieve the state of such signals from slaves. Strictly speaking this behavior is not covered by AMBA3, but is provided with the AMBA-PV components.

`set_state(int export_id, const bool &state): void`
Transfers a signal state. The `export_id` parameter must be set to 0 in this context.

`get_state(int export_id, tlm::tlm_tag<bool> * t): bool`
Retrieves a signal state. The `export_id` parameter must be set to 0, and the `t` parameter must be set to `NULL`, in this context.

2.1.6 AMBAPVValue protocol

This protocol models propagation of 32-bit integer values between components. Strictly speaking this behavior is not covered by AMBA3, but is provided with the AMBA-PV components.

`set_state(int export_id, const uint32_t & value): void`
Transfers a value. The `export_id` parameter must be set to 0 in this context.

2.1.7 AMBAPVValue64 protocol

This protocol models propagation of 64-bit integer values between components. Strictly speaking this behavior is not covered by AMBA3, but is provided with the AMBA-PV components.

`set_state(int export_id, const uint64_t &value): void`
Transfers a value. The `export_id` parameter must be set to 0 in this context.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

2.1.8 AMBAPVValueState protocol

This protocol permits propagation of 32-bit integer values between components and their retrieval from slaves. Strictly speaking this behavior is not covered by AMBA3, but is provided with the AMBA-PV components.

`set_state(int export_id, const uint32_t &value):void`
Transfers a value. The `export_id` parameter must be set to 0 in this context.

`get_state(unsigned int export_id, tlm::tlm_tag<uint32_t> *t): uint32_t`
Retrieves a value. The `export_id` parameter must be set to 0, and the `t` parameter must be set to `NULL`, in this context.

2.1.9 AMBAPVValueState64 protocol

This protocol permits propagation of 64-bit integer values between components and their retrieval from slaves.

Strictly speaking this behavior is not covered by AMBA3, but is provided with the AMBA-PV components.

`set_state(int export_id, const uint64_t &value): void`
Transfers a value. The `export_id` parameter must be set to 0 in this context.

`get_state(int export_id, tlm::tlm_tag<uint64_t> *t): uint64_t`
Retrieves a value. The `export_id` parameter must be set to 0, and the `t` parameter must be set to `NULL`, in this context.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

2.2 Clocking protocols

This section describes the clocking protocols.

This section contains the following subsections:

- *2.2.1 Clocking protocols - about* on page 2-65.
- *2.2.2 ClockRateControl protocol* on page 2-65.
- *2.2.3 TimerCallback protocol* on page 2-65.
- *2.2.4 TimerCallback64 protocol* on page 2-65.
- *2.2.5 TimerControl protocol* on page 2-65.
- *2.2.6 TimerControl64 protocol* on page 2-66.

2.2.1 Clocking protocols - about

The clocking components and protocols provide a mechanism for systems to regulate the execution rate of components.

All clocking components that communicate use an opaque ClockSignal protocol. ClockSignal protocols have no behaviors that the user can invoke.

Related references

3.3 Clocking components on page 3-113

2.2.2 ClockRateControl protocol

This protocol sets the ratio of this component.

`set(uint32_t mul, uint32_t div) : void`

`set64(uint64_t mul, uint64_t div) : void`

Set the multiplier and divider that determine the clock divider ratio.

$clk_out_{freq} = clk_in_{freq} * mul / div$

2.2.3 TimerCallback protocol

This protocol invokes a behavior on the component that set the timer. Do not use the `timer_control` port of the timer during the invoked behavior.

`signal() : uint32_t`
Invoked by the timer when the timer countdown expires. The invoked behavior returns the number of ticks after which it is called again or 0 to make the timer one-shot.

2.2.4 TimerCallback64 protocol

This protocol invokes a behavior on the component that set the timer. Do not use the `timer_control` port of the timer during the invoked behavior.

`signal() : uint64_t`
Invoked by the timer when the timer countdown expires. The invoked behavior returns the number of ticks after which it is called again or 0 to make the timer one-shot.

2.2.5 TimerControl protocol

This protocol controls the actions of the component. It permits a timer to be set to schedule a callback after a given number of ticks at the rate of the clock input.

If a timer is set while it is counting, it starts counting the new number of ticks without sending the original callback. Canceling a timer when it is not active has no effect.

`set(uint32_t ticks) : void`
Set the timer to countdown the given number of ticks.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

`cancel()`
Cancel an active timer, preventing the callback being invoked.

`isSet() : bool`
Check whether a timer is set to generate a callback.

`remaining() : uint32_t`
Return how many ticks remain before the callback is invoked.

2.2.6 TimerControl64 protocol

This protocol controls the actions of the component. It permits a timer to be set to schedule a callback after a given number of ticks at the rate of the clock input.

If a timer is set while it is counting, it starts counting the new number of ticks without sending the original callback. Canceling a timer when it is not active has no effect.

`set(uint64_t ticks) : void`
Set the timer to countdown the given number of ticks.

`cancel()`
Cancel an active timer, preventing the callback being invoked.

`isSet() : bool`
Check whether a timer is set to generate a callback.

`remaining() : uint64_t`
Return how many ticks remain before the callback is invoked.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

2.3 Debug interface protocols

This section describes the debug interface protocols.

This section contains the following subsections:

- *2.3.1 Debug interface protocols - about* on page 2-67.
- *2.3.2 CADIDisassemblerProtocol protocol* on page 2-67.
- *2.3.3 CADIProtocol protocol* on page 2-68.

2.3.1 Debug interface protocols - about

LISA components can expose aspects of their internal state so that they become visible and usable in the debugger. Some aspects are supported by the native LISA language, and some are supported by debug interface protocols.

Registers and other state variables
See the LISA+ resource REGISTER declaration, especially the `read_function` and `write_function` attributes.

Memories and other memory-like objects
See the LISA+ resource MEMORY declaration, especially the `read_function` and `write_function` attributes.

Disassembly
See CADIProtocol and CADIDisassemblerProtocol.

Instruction count
See CADIProtocol.

Setting, configuring and clearing of breakpoints
For example, code breakpoints, register breakpoints, watchpoints. See CADIProtocol.

Single stepping and instruction stepping support.
See CADIProtocol.

If displaying and editing memory and registers are sufficient, you need not implement CADIProtocol and CADIDisassemblerProtocol. The component must implement them to enable the other debug features.

CADIProtocol and CADIDisassemblerProtocol permit you to implement the features on a *Component Architecture Debug Interface* (CADI) interface level, where the component code takes responsibility for the implementation of these interfaces.

Related information

Component Architecture Debug Interface Developer Guide v2.0

LISA+ Language for Fast Models Reference Manual

2.3.2 CADIDisassemblerProtocol protocol

To support disassembly, implement all of these functions. None of them is optional.

These functions are in a different port, of type CADIDisassemblerProtocol, that can have any name and only need to be implemented when disassembly must be exposed in the debugger. The functionality of this port is then exposed by `CADIProtocol::CADIGetDisassembler()`. See CADIProtocol for information on how to use this port and `CADIDisassemblerAdapter`.

In the function `slave behavior GetType(): eslapi::CADIDisassemblerType;` components must always return `eslapi::CADI_DISASSEMBLER_TYPE_STANDARD`.

The function `slave behavior GetModeCount(): uint32_t;` returns the number of supported disassembler modes, and at least 1 mode must be returned.

The function `slave behavior GetModeNames(eslapi::CADIDisassemblerCB *callback_);` returns information about all supported modes. A component that only supports one mode calls, for example,

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

`callback_->ReceiveModeName(0, "Normal");` only once. This is similar for multiple modes with different names and Ids.

The function `slave behavior GetCurrentMode(): uint32_t;` returns the most suitable mode of disassembly at the time, based on the current state variables of the component:

For the function:

```
slave behavior GetSourceReferenceForAddress(eslapi::CADIDisassemblerCB *callback_,
const eslapi::CADIAddr_t &address): eslapi::CADIDisassemblerStatus;
```

components must return `eslapi::CADI_DISASSEMBLER_STATUS_ERROR;`

For the function:

```
slave behavior GetAddressForSourceReference(const char *sourceFile, uint32_t
sourceLine, eslapi::CADIAddr_t &address): eslapi::CADIDisassemblerStatus;
```

components must return `eslapi::CADI_DISASSEMBLER_STATUS_ERROR;`

The following function is the main disassembler function:

```
slave behavior GetDisassembly(eslapi::CADIDisassemblerCB *callback_,
                              const eslapi::CADIAddr_t &address,
                              eslapi::CADIAddr_t &nextAddr,
                              const uint32_t mode,
                              uint32_t desiredCount):
                              eslapi::CADIDisassemblerStatus;
```

The component must call `callback_` for all disassembler lines for the specified address and `desiredCount`, and it must finally set `nextAddr` to the next disassembled address at that point after the requested block.

```
// Query if an instruction is a call instruction
    slave behavior GetInstructionType(const eslapi::CADIAddr_t &address,
eslapi::CADIDisassemblerInstructionType &insn_type): eslapi::CADIDisassemblerStatus;
```

Components must return `insn_type = eslapi::CADI_DISASSEMBLER_INSTRUCTION_TYPE_NOCALL;` and return `eslapi::CADI_DISASSEMBLER_STATUS_OK;`

Related references

2.3.3 CADIProtocol protocol on page 2-68

Related information

Component Architecture Debug Interface Developer Guide Version 2.0

2.3.3 CADIProtocol protocol

This protocol supports debugging. To add breakpoint support, implement `CADIBpt...()` functions.

Note

All CADIProtocol protocol behaviors, that is, all sets of functionalities, are optional. A component only has to implement the set of functions for the functionality that it intends to support.

You must define an internal slave port of this type, and the name of this port must always be `cadi_port`. In this port, you can implement this functionality:

```
optional slave behavior CADIBptGetList(uint32_t, uint32_t, uint32_t *, eslapi::CADIBptDescription_t
*):eslapi::CADIReturn_t;
optional slave behavior CADIBptRead(eslapi::CADIBptNumber_t, eslapi::CADIBptRequest_t *):eslapi::CADIReturn_t;
optional slave behavior CADIBptSet(eslapi::CADIBptRequest_t *, eslapi::CADIBptNumber_t
*):eslapi::CADIReturn_t;
optional slave behavior CADIBptClear(eslapi::CADIBptNumber_t):eslapi::CADIReturn_t;
optional slave behavior CADIBptConfigure(eslapi::CADIBptNumber_t,
eslapi::CADIBptConfigure_t):eslapi::CADIReturn_t;
optional slave behavior CADIModifyTargetFeatures(eslapi::CADITargetFeatures_t *):eslapi::CADIReturn_t;
```


Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

You need to implement all of these functions to enable any kind of breakpoint for the component. The component needs to maintain and keep track of all existing breakpoints. For example:

- `CADIBptSet()` and `CADIBptClear()` add and remove breakpoints.
- `CADIBptConfigure()` enable and disable breakpoints.
- `CADIBptGetList()` and `CADIBptRead()` return the current state of the breakpoint list.

The component must also implement `CADIModifyTargetFeatures`. This function permits you to override the automatic default `CADITargetFeatures_t` that System Generator provides for this component just before it is returned to the debugger. Specifically, a component that wants to support any kind of breakpoint must override the `handledBreakpoints` and `nrBreakpointsAvailable` fields of `CADITargetFeatures_t`. For example:

```
targetFeatures->handledBreakpoints = CADI_TARGET_FEATURE_BPT_PROGRAM | CADI_TARGET_FEATURE_BPT_REGISTER;
    // code and register breakpoints supported
targetFeatures->nrBreakpointsAvailable = 0x7fffffff;
    // virtually infinite number of breakpoints supported.
```

By default, LISA components do not support any type of breakpoints. By implementing `CADIBpt...()` functions, you can add breakpoint support. In addition to implementing the stated functions, the component must call `simBreakpointHit(bptNumber)` and then `simHalt()` when an enabled breakpoint is hit. On a breakpoint hit the component must first call `simBreakpointHit()` for each breakpoint that was hit (one or more, usually just one) and then call `simHalt()` once after all `simBreakpointHit()` calls. The `simHalt()` call must always be the last call in the sequence.

A component that wants to provide disassembly must implement the following `CADIGetDisassembler()` behavior and return a CADIDisassembler interface implementation. This automatically follows behind the `CADI::CADIGetDisassembler()` and the `CADI::ObtainInterface("eslapi.CADIDisassembler2")` functions.

```
optional slave behavior CADIGetDisassembler():eslapi::CADIDisassembler*;
```

To do this, instantiate a `CADIDisassemblerAdapter` object in behavior `init` and return its address in this `CADIGetDisassembler()` function. This object must point to an internal slave port that implements the CADIDisassemblerProtocol protocol.

Skeleton code for implementing disassembly

```
component FOO
{
    behavior init()
    {
        disassemblerAdapter = new CADIDisassemblerAdapter(disassPort.getAbstractInterface());
        // ...
    }
    internal slave port <CADIProtocol> cadi_port
    {
        slave behavior CADIGetDisassembler():eslapi::CADIDisassembler*
        {
            return disassemblerAdapter;
        }
        // ...
    }
    internal slave port<CADIDisassemblerProtocol> disassPort
    {
        // ...
    }
}
```

The following function implements the instruction *stepping a component*. It must set up an internal state that stops the simulation when the requested number of instructions is executed completely (exactly like a breakpoint). It must call `simRun()` from within `CADIExecSingleStep()` after setting up this stepping state, and later it must call `simHalt()` when the execution of the required number of instructions finishes.

```
optional slave behavior CADIExecSingleStep(uint32_t instructionCount, int8_t stepCycle, int8_t
stepOver):eslapi::CADIReturn_t;
```


Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

The following function is for debugging purposes only. Do not implement it. The function must not alter the state of any component in any way.

```
optional slave behavior callbackModeChange(uint32_t newMode, eslapi::CADIBptNumber_t bptNumber);
```

By implementing any of the following functions, the component can enable the instruction and cycle count display:

```
optional slave behavior CADIGetInstructionCount(uint64_t &instructionCount):eslapi::CADIReturn_t;
optional slave behavior CADIGetCycleCount(uint64_t &instructionCount, bool systemCycles):eslapi::CADIReturn_t;
```

——— **Note** ———

Fast Models systems are not cycle accurate, so you usually only implement an instruction counter, if at all.

———————————

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

2.4 Peripheral protocols

This section describes the peripheral protocols.

This section contains the following subsections:

- *2.4.1 AudioControl protocol* on page 2-71.
- *2.4.2 CharacterLCD protocol* on page 2-71.
- *2.4.3 FlashLoaderPort protocol* on page 2-71.
- *2.4.4 GUIPollCallback protocol* on page 2-71.
- *2.4.5 ICS307Configuration protocol* on page 2-72.
- *2.4.6 KeyboardStatus protocol* on page 2-72.
- *2.4.7 LCD protocol* on page 2-72.
- *2.4.8 LCDLayoutInfo protocol* on page 2-73.
- *2.4.9 MMC_Protocol protocol* on page 2-73.
- *2.4.10 MouseStatus protocol* on page 2-74.
- *2.4.11 PL080_DMAC_DmaPortProtocol protocol* on page 2-74.
- *2.4.12 PS2Data protocol* on page 2-74.
- *2.4.13 PVBusSlaveControl protocol* on page 2-75.
- *2.4.14 PVDevice protocol* on page 2-77.
- *2.4.15 PVTransactionMaster protocol* on page 2-77.
- *2.4.16 SerialData protocol* on page 2-77.
- *2.4.17 SMMUv3AEMIdentifyProtocol protocol* on page 2-78.
- *2.4.18 TZFilterControl protocol* on page 2-79.
- *2.4.19 VirtualEthernet protocol* on page 2-79.

2.4.1 AudioControl protocol

This protocol has get and release audio buffer behaviors.

`getPVAudioBuffer`
Get an underlying host buffer for audio output.

`releasePVAudioBuffer`
Release an underlying host buffer.

2.4.2 CharacterLCD protocol

This protocol has the behaviors `setLayoutInfo` and `draw`.

`setLayoutInfo`
sets the width and height of the touchscreen.

`draw`
sets the character information.

2.4.3 FlashLoaderPort protocol

This protocol initializes the flash contents at model startup and saves flash contents to a file when the model terminates.

`loadFlashFile(FlashLoader *) : uint32`
Initiate loading of the flash contents.

`saveFlashFile(FlashLoader *) : uint32`
Save the flash contents to a file.

2.4.4 GUIPollCallback protocol

This protocol defines one method that signals to the visualization component the end of the update period. You can invoke this callback even when the simulation is stopped.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

`gui_callback() : void`
This is sent by the GUIPoll component, at the configured period.

2.4.5 ICS307Configuration protocol

This protocol sets the divider ratio of an ICS307 component at runtime. The output clock rate alters accordingly and any dependent components react to the clock rate change according to their defined behavior.

`setConfiguration(uint32_t vdw, uint32_t rdw,uint32_t od): void`

Set the parameters for deriving the clock divider ratio.

vdw

Range: 0-255.

rdw

Range: 0-255.

od

Range: 0-7.

2.4.6 KeyboardStatus protocol

This protocol passes keyboard events to a component such as the PS2Keyboard component.

Events are only sent when the visualization window is in focus. Keyboard combinations that are filtered by the host OS such as **Ctrl+Alt+Del** are not detected by the visualization. See `components/KeyCode.h` for a list of ATKeyCode `code` values.

The protocol behaviors are:

`keyDown(ATKeyCode code) : void`
This is sent when a key on the host keyboard is pressed.

`keyUp(ATKeyCode code) : void`
This is sent when a key on the host keyboard is released.

Related references
VEVisualisation - ports on page 9-3012

2.4.7 LCD protocol

This Visualisation Library signaling protocol provides the interface between an LCD controller peripheral (for example the PL110) and a visualization component. This permits the LCD controller to render the framebuffer contents into a region of the visualization GUI.

LISA visualization components can provide any number of LCD ports. The implementations of these behaviors can delegate the calls to appropriate methods on the VisRenderRegion class.

The behaviors are:

`lock() : VisRasterLayout *`
Lock the raster region of the LCD in preparation for rendering.

`unlock()`
Unlock the raster region, ready to be updated on the screen.

`update(int x, int y, unsigned int w, unsigned int h)`
Update the selected rectangular area on screen from the raster buffer.

`setPreferredLayout(unsigned int width, unsigned int height, unsigned int depth)`
A request from the LCD controller to set the preferred size for the raster region, to match the timing parameters used by the LCD controller.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Related references

VEVisualisation - ports on page 9-3012

2.4.8 LCDLayoutInfo protocol

This protocol has the behavior `setLayoutInfo`.

`setLayoutInfo`
sets the width and height of the touchscreen.

2.4.9 MMC_Protocol protocol

This protocol describes an abstract, untimed interface between an MMC controller and an MMC or SD card.

The protocol contains methods that must be implemented by the master (controller) and some that must be implemented by the slave (card). This protocol is used by the reference PL180 MCI and MMC models. for further information on the protocol implementation, see the source file, `%PVLIB_HOME%\LISA\MMC_Protocol.lisa`.

Use of this protocol assumes knowledge of the MultiMediaCard specification, available from the MultiMediaCard Association, `www.mmca.org`.

The protocol has these behaviors:

`cmd`
Commands are sent from the controller to the card using this behavior, which is implemented by the card model. The MMC command is sent with an optional argument. The card responds as defined by the MMC specification. The controller model checks that the response type matches expectations, and updates its state appropriately. The transaction-level protocol does not model start/stop bits or CRCs on the command/response payload.

For data transfer in the card to controller direction:

`Rx`
After the host and controller have initiated a read through the command interface, the card calls the `Rx` behavior on the controller to provide the read data. The call provides a pointer and a length. The ARM MMC reference model simulates device read latency by waiting a number of clock cycles prior to calling this behavior. If the controller is unable to accept the data, or wants to force a protocol error, it can return false in response to this behavior.

`Rx_rdy`
A handshake, used by the controller to inform the card that the controller is ready to receive more data. The ARM MMC reference model does not time out, so waits indefinitely for this handshake in a multiple block data transfer.

For data transfer in the controller to card direction:

`Tx`
After the host and controller have initiated a write through the command interface, the card calls the `Tx` behavior on the controller. The call provides a pointer to an empty buffer to be written, and a length. The ARM MMC reference model simulates device write latency by waiting a number of clock cycles prior to each buffer being offered.

`Tx_done`
The controller calls this behavior on the card when the block has been written. The card model can then commit the data to its persistent storage.

The card model must also implement:

`cmd_name`
This behavior returns the name of the command issued. A card must implement this behavior, but is free to return an empty string for all requests. Only call this behavior for diagnostic messages.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

2.4.10 MouseStatus protocol

This protocol passes mouse movement and button events to another component such as the PS2Mouse component.

Events are only sent when the visualization window is in focus.

The protocol behaviors are:

`mouseMove(int dx, int dy) : void`
This is sent when the host mouse is moved. Mouse movement events are always relative.

`mouseButton(uint8_t button, bool down) : void`

This is sent when a button on the host mouse is pressed or released.

`button` indicates which button has been pressed or released and is typically 0, 1, or 2 but can be anything up to 7 depending on the OS and attached mouse.

`down` is true if a button is pressed and false if released.

Related references
VEVisualisation - ports on page 9-3012

2.4.11 PL080_DMAC_DmaPortProtocol protocol

This protocol provides methods to permit handshaking between peripherals and the DMA controller.

`request(uint32 request) : void`

Passes requests from a peripheral to the DMA controller. The request is a bitfield with the low four bits defined. The request is level sensitive and latched internally by the DMA controller. It is sampled and interpreted in a manner dependent on the target channel and configured flow control.

`0: PL080_REQ_BURST`

Burst transfer request.

`1: PL080_REQ_SINGLE`
Single transfer request.

`2: PL080_REQ_LBURST`
Last burst request.

`3: PL080_REQ_LSINGLE`
Last single request.

`response(uint32 response) : void`

Passes responses from the DMA controller to peripherals. The response is a bitfield with the low two bits defined. The response is transient rather than level sensitive.

`0: PL080_RES_TC`

Terminal count response.

`1: PL080_RES_CLR`
Clear request response.

2.4.12 PS2Data protocol

This protocol is for communication between the KMI and a PS/2-like device.

For efficiency, the interface is a parallel byte interface rather than a serial clock/data interface. The behaviors are:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

setClockData(enum ps2clockdata) : void
Used by the KMI to simulate forcing the state of the data/clock lines, to indicate whether it is able to receive data, wants to send a command, or is inhibiting communication.

getData() : uint8
Used by the PS/2 device to get command data from the KMI.

putData(uint8 data) : void
Used by the PS/2 device to send device data to the KMI.

2.4.13 PVBusSlaveControl protocol

The PVBusSlaveControl protocol enables you to access and modify the underlying memory that PVBusSlave controls.

setFillPattern(uint32_t fill1, uint32_t fill2)
This sets a two-word alternating fill pattern to be used by uninitialized memory.

setAccess(pv::bus_addr_t base, pv::bus_addr_t top, pv::accessType type, pv::accessMode mode)

This reconfigures handling for a region of memory.

`base` (inclusive value) and `top` (exclusive value) specify the address range to configure.

`type` selects what types of bus access must be reconfigured. It can be one of:

- `pv::ACCESSTYPE_READ`
- `pv::ACCESSTYPE_WRITE`
- `pv::ACCESSTYPE_RW`

`mode` controls what happens when an address is accessed. Legal values for the `pv::accessMode` enumeration are:

pv::ACCESSMODE_MEMORY
Act as memory. The PVBusSlave manages the underlying storage to provide 4KB of memory, which can be ROM or RAM, depending on how you configure it to handle bus write transactions.

pv::ACCESSMODE_DEVICE
Act as a device. Requests to the select pages are routed to the PVBusSlave `device` port, where the necessary behavior can be implemented by the component.

pv::ACCESSMODE_ABORT
Generate bus abort signals for any accesses to this page.

pv::ACCESSMODE_IGNORE
Ignore accesses to this page. Bus read requests return 0.

getReadStorage(pv::bus_addr_t address, pv::bus_addr_t *limit) : const uint8_t *

getWriteStorage(pv::bus_addr_t address, pv::bus_addr_t *limit) : uint8_t *

These two methods permit you to access the underlying storage that PVBusSlave allocates to implement a region of memory.

The return value is a pointer to the byte that represents the storage corresponding to the address of `base`.

The limit pointer returns the device address for the limit of the accessible memory.

The pointer value returned is not guaranteed to remain valid indefinitely. Bus activities, or other calls to the control port, might invalidate the pointer.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

provideReadStorage(pv::bus_addr_t device_base, pv::bus_addr_t device_limit, const uint8_t *storage)

provideWriteStorage(pv::bus_addr_t device_base, pv::bus_addr_t device_limit, uint8_t *storage)

provideReadWriteStorage(pv::bus_addr_t device_base, pv::bus_addr_t device_limit, uint8_t *storage)

These methods enable you to allocate blocks of memory that the PVBusSlave can use to manage regions of RAM/ROM. Only use these methods when you require a high degree of control over memory, such as when you require a device to map specific regions of host memory into the simulation.

The memory region pointed to by `storage` must be large enough to contain `(limit - base)` bytes.

After these calls, PVBusSlave controls access to the underlying memory. The owner must call `getWriteStorage()` before modifying the memory contents and `getReadStorage()` before reading the memory contents.

provideReadStorageEx(pv::bus_addr_t device_base, pv::bus_addr_t device_limit, const uint8_t *storage, double latency)

provideWriteStorageEx(pv::bus_addr_t device_base, pv::bus_addr_t device_limit, const uint8_t *storage, double latency)

provideReadWriteStorageEx(pv::bus_addr_t device_base, pv::bus_addr_t device_limit, uint8_t *storage, double read_latency, double write_latency)

These methods take additional parameters to specify average latencies (in seconds) per byte, used when Timing Annotation is enabled.

In all other aspects they behave the same as `provideReadStorage()`, `provideWriteStorage()` and `provideReadWriteStorage()`, respectively.

slave behavior getRegionIterHandle(): uint32_t;

slave behavior getNextRegionInfo(uint32_t iter_handle, pv::PVBusSlaveRegionInfo *info) : bool;

slave behavior closeRegionIterHandle(uint32_t iter_handle);

These methods form an iterator-like API that allows a PVBusSlave providing storage to report all the regions of the address space that have backing store.

The iteration begins by calling `getRegionIterHandle()`. This allocates an iterator and if successful returns a nonzero `iter_handle` to identify it.

The caller can then repeatedly call `getNextRegionInfo()` with `iter_handle`. If it finds a region, the behavior returns `true` and writes to the `info` struct if the pointer is non-null. Access the data in the region using `getReadStorage()` or `getWriteStorage()`.

The implementation can return regions in any order. They can be of any size or alignment, but must not overlap.

The implementation need not report allocated regions that are filled entirely with the default fill pattern, or allocated regions that contain only the data they had at simulation start.

On reaching the last region, the iterator closes automatically. If the handle is invalid or there are no further regions, the behavior returns `false`.

A caller can close an iterator opened by `getRegionIterHandle()` at any time using `closeRegionIterHandle()`. This deallocates the iterator, and further uses of the handle are invalid.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

2.4.14 PVDevice protocol

The PVDevice protocol enables you to implement support for memory-mapped device registers. Call the two methods through the device port on the PVBusSlave to handle bus read/write transactions.

`read(pv::ReadTransaction) : pv::Tx_Result`
This method permits a device to handle a bus read transaction.

`write(pv::WriteTransaction) : pv::Tx_Result`
This method permits a device to handle a bus write transaction.

The PVDevice protocol uses two behaviors to differentiate between transactions originating from the processor (loads and stores) and transactions originating from an attached debugger:

`slave behavior debugRead(pv::ReadTransaction tx) : pv::Tx_Result`
This method enables the device to handle a debug read transaction.

`slave behavior debugWrite(pv::WriteTransaction tx) : pv::Tx_Result`
This method enables the device to handle a debug write transaction.

The `debugRead` and `debugWrite` behaviors are called for all debug transactions.

For an example component that uses this protocol, see `$PVLIB_HOME/examples/LISA/BusComponents/BitLatch.lisa`.

Related references
1.15 PVBus C++ transaction and Tx_Result classes on page 1-50

2.4.15 PVTransactionMaster protocol

This protocol instantiates a `pv::TransactionGenerator` object from a PVBusMaster.

`createTransactionGenerator( ) : pv::TransactionGenerator *`

This behavior instantiates a new transaction generator to control the bus master. A caller can allocate as many TransactionGenerators as it wants. It is up to the caller to delete TransactionGenerators when they are no longer required. For example:

```
behavior init() {
    tg = master.createTransactionGenerator();
}

behavior terminate() {
delete tg;
}
```

2.4.16 SerialData protocol

This protocol is implemented as a parallel interface for efficiency. All communication is driven by the master port.

This protocol has behaviors:

`dataTransmit(uint16_t data) : void`

Used by the master to send data to the slave.

Table 2-1 Bits for dataTransmit()

Bits	Function
15:8	Reserved
7:0	Transmit data

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

`dataReceive(void) : uint16_t`

Used by the master to receive data from the slave.

Table 2-2 Bits for dataReceive()

Bits	Function
15:13	Reserved
12	Set when no data available for reading
11	Reserved
10	Break error
9:8	Reserved
7:0	Receive data

`signalsSet(uint8_t signal) : void`

Used by the master to get the current signal status.

Table 2-3 Bits for signalsSet()

Bits	Function
7	Out1
6	Out2
5	RTS
4	DTR
3:0	Reserved

`signalsGet() : uint8_t`

Used by the master to get the current signal status.

Table 2-4 Bits for signalsGet()

Bits	Function
7:4	Reserved
3	DCD
2	DSR
1	CTS
0	RI

2.4.17 SMMUv3AEMIdentifyProtocol protocol

This protocol is used by the SMMUv3AEM model to identify the SSD, StreamID, and SubStreamID of an incoming transaction. It is only used if the parameter `howto_identify` is set to `"use-identify"`.

```
identify(unsigned tbu_number_, const pv::TransactionAttributes* attributes_, bool*
out_ssd_ns_, unsigned* out_streamid_, unsigned* out_substreamid_) : void
```

Architecturally, a transaction comes into the SMMUv3AEM model with the following side band signals:

- Security State Determination (SSD):

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

0
Transaction belongs to a device controlled by the secure world.

1
Transaction belongs to a device controlled by the non-secure world.

- StreamID.
- SubStreamID and SubStreamID valid.

  If you set `*out_substreamid_ = ~0u`, that is interpreted as no SubStreamID because SubStreamIDs are 20 or fewer bits.

How these are transported in the system is SoC-dependent.

The SMMUv3AEM requires that the SoC provides a way of determining this information by implementing the `identify()` function.

2.4.18 TZFilterControl protocol

This protocol controls the communication between filter units and control registers in the APB control block.

`checkPermission(const pv::TransactionAttributes *attributes_, pv::bus_addr_t page_base_, bool is_read_, pv::RemapRequest &req_, bool &abort_on_error_) : bool`
Check the permission of the transactions filtered by the filter unit. Optional slave behavior.

`isEnabled() : bool`
Check if the filter unit is enabled or not. The APB control block controls the unit. Slave behavior.

`isSecureSlave() : bool`
Check if the connected slave is secure or not. Optional slave behavior.

`setConfig(bool rd_spec_enable, bool wr_spec_enable, uint32_t action) : void`
Pass the configurations to the filter. Optional master behavior.

2.4.19 VirtualEthernet protocol

This protocol has the sendToSlave and sendToMaster behaviors.

`sendToSlave(EthernetFrame *frame)`
Send an Ethernet frame to the slave port.

`sendToMaster(EthernetFrame *frame)`
Send an Ethernet frame to the master port.

The Ethernet frame class encapsulates an Ethernet frame in a broken-up format that is more accessible by components. For information on the class definition, see the `EthernetFrame.h` header file located in `$PVLIB_HOME/include/components/VirtualEthernet/Protocol`.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

2.5 Power management protocols

This section describes the power management protocols.

This section contains the following subsection:

- *2.5.1 PChannel protocol* on page 2-80.

2.5.1 PChannel protocol

Communicates power state changes between a power controller and a device.

The behaviors of the protocol are:

`pactive (uint32_t pstate) : void`

This master behavior is implemented by a power controller. A device calls this method to give a hint to the power controller that it can change to a particular power state. A power controller can then take appropriate action, typically communicating with the device by calling `device.prequest(new_power_state)`.

The power state is type `uint32_t` because it is the responsibility of the system using PChannels to enumerate the power states that it supports. For example, ARMv8-A cores use the following enumeration for power states:

```
enum { OFF = 0, OFF_EMU, MEM_RET, MEM_RET_EMU, LOGIC_RET, FULL_RET, MEM_OFF,
FUNC_RET, ON, WARM_RST, DBG_RECOV }
```

`prequest (uint32_t pstate) : sg::PChannel::presp_t`

This slave behavior is implemented by a device, for instance a core. A power controller typically calls this method and checks for the response from the device, which can either be `ACCEPT` or `DENY`.

`sg::PChannel::presp_t`

This enumeration provides two values, `ACCEPT` and `DENY`. It is returned by the `prequest()` method, depending on the state requested and the current state of the core.

Usage

You can use PChannels to replace `STANDBYWFI` and `STANDBYWFE` signaling.

For example, using `STANDBYWFI` or `STANDBYWFE`:

- Core drives `STANDBYWFI` signal HIGH.
- Power controller performs logic *x*.

Equivalent behavior using PChannels:

- Core calls `pactive(OFF)`.
- Power controller calls `prequest(OFF)` to change the core to OFF.
- Power controller performs logic *x*.
- To wake up the core, the power controller calls `prequest(ON)`.

Note

- For a LISA+ example that demonstrates how to use PChannel, see `$PVLIB_HOME/examples/LISA/VP_PChannel/`.
- For SystemC test code that demonstrates how to use PChannel, see `$PVLIB_HOME/Platforms/SystemCExport/EVS_Components/EVS_PChannel/`.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

2.6 Processor protocols

This section describes the processor protocols.

This section contains the following subsections:

- *2.6.1 CoprocBusProtocol protocol* on page 2-81.
- *2.6.2 CounterInterface protocol* on page 2-84.
- *2.6.3 GICv3Comms protocol* on page 2-84.
- *2.6.4 InstructionCount protocol* on page 2-84.
- *2.6.5 v8EmbeddedCrossTrigger_controlprotocol protocol* on page 2-84.

2.6.1 CoprocBusProtocol protocol

This protocol connects a coprocessor implementation with a CPU component, for instance ARMCortexM33CT.

A coprocessor must derive from the coprocessor callback interface, `Coprocessor`. It can implement the `CDP`, `MCR`, `MRC`, `STC`, `LDC`, `MCRR`, and `MRRC` instructions.

A coprocessor must be registered with a specific coprocessor number, by calling the `addCoprocessor()` method. You can only register an external coprocessor that is not already present in the CPU. If no coprocessor has been registered with the coprocessor number encoded in an instruction, the CPU raises a NOCP fault.

To register coprocessor instruction implementations with the CPU, you must initialize the function pointers. For example, the following code passes the function pointers to the `Coprocessor` constructor. This code was taken from the `$PVLIB_HOME/examples/LISA/FVP_Coproc_Demo` example.

Registering a coprocessor

```
…
    class TestValCoprocessor : public Coprocessor
    {
    public:
        protocol_CoprocBusProtocol * coproc_bus;
        uint32_t coproc_number;
        uint32_t cp_reg[2][NUM_CP_REG] = {{0}}; // [0][NUM_CP_REG] --> Secure, [1][NUM_CP_REG] --> Non-Secure
        TestValCoprocessor()
        : Coprocessor(this, test_CDP, nullptr, test_MCR, nullptr, test_MRC, nullptr, test_LDC, nullptr,
test_STC, nullptr, test_MCRR, nullptr, test_MRRC, nullptr)
        , coproc_bus(nullptr)
        , coproc_number(0)
        {
        }
…
    };

    PARAMETER { description("coprocessors number"), type(uint32_t), default(0x2), min(0x0), max(16) }
coprocessor_number; // CP num
    TestValCoprocessor test_cp;
}

behaviour init
{
…
    if (coproc_bus.addCoprocessor.implemented())
    {
        coproc_bus.addCoprocessor(&test_cp, coprocessor_number);
    }
}
```

Coprocessor behaviors

A coprocessor can call the following master behaviors:

`addCoprocessor(Coprocessor*, int num) : void`
Registers the coprocessor with the CPU. `num` identifies which coprocessor to register it as.

`removeCoprocessor(Coprocessor*, int num) : void`
Unregisters the coprocessor from the CPU.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

`accessIsPriv(void) : bool`
Checks whether the CPU state is privileged (true) or unprivileged (false).

`accessIsNonSecure(void) : bool`
Checks the security state of the CPU, either true for non-secure, or false for secure.

Coprocessor callback functions

A coprocessor can implement callback functions with the following signatures.

Each function returns a `CoprocState` value to indicate the new transaction state of the coprocessor, as listed in *CoprocState values* on page 2-83.

- Perform a coprocessor data processing operation.

```
CoprocState CDP(void* context, uint32_t inst)
```

Parameters:

`context`
context that was registered with the coprocessor interface.
`inst`
the coprocessor instruction being executed.

- Perform a move to coprocessor register operation.

```
CoprocState MCR(void* context, uint32_t inst, uint32_t data)
```

Parameters:

`context`
context that was registered with the coprocessor interface.
`inst`
the coprocessor instruction being executed.
`data`
register contents.

- Perform a move from coprocessor register operation.

```
CoprocState MRC(void* context, uint32_t inst, uint32_t* data)
```

Parameters:

`context`
context that was registered with the coprocessor interface.
`inst`
the coprocessor instruction being executed.
`data`
pointer to word to fill with coprocessor register contents.

- Perform a load coprocessor register from memory operation.

```
CoprocState LDC(void* context, uint32_t inst, uint32_t data, CoprocState state)
```

Parameters:

`context`
context that was registered with the coprocessor interface.
`inst`
the coprocessor instruction being executed.
`data`
contents of current memory location to load into register.
`state`
current state in a sequence of transactions.

- Perform a store coprocessor register to memory operation.

```
CoprocState STC(void* context, uint32_t inst, uint32_t data, CoprocState state)
```


Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Parameters:

context
: context that was registered with the coprocessor interface.

inst
: the coprocessor instruction being executed.

data
: pointer to word to fill with coprocessor register contents to be transferred to memory.

state
: current state in a sequence of transactions.

- Perform a move to two coprocessor registers operation.

```
CoprocState MCRR(void* context, uint32_t inst, uint32_t data1, uint32_t data2)
```

Parameters:

context
: context that was registered with the coprocessor interface.

inst
: the coprocessor instruction being executed.

data1
: first data word to load to a coprocessor register.

data2
: second data word to load to a coprocessor register.

- Perform a move from two coprocessor registers operation.

```
CoprocState MRRC(void* context, uint32_t inst, uint32_t* data1, uint32_t* data2)
```

Parameters:

context
: context that was registered with the coprocessor interface.

inst
: the coprocessor instruction being executed.

data1
: pointer to first word to fill with coprocessor register contents.

data2
: pointer to second word to fill with coprocessor register contents.

CoprocState values

A `CoprocState` enum value is returned by coprocessor callback functions to indicate the new transaction state of the coprocessor. It is also used as a parameter for `LDC` and `STC` callback functions.

Table 2-5 CoprocState values

Value	State label	Description
`0x0`	`CoprocOk`	Complete/Ok.
`0x1`	`CoprocUndef`	Undefined operation.
`0x2`	`CoprocAbort`	Data abort.
`0x4`	`CoprocFirst`	A parameter value for `LDC` and `STC` callback functions to indicate that this is the first data transfer in a sequence.
`0x5`	`CoprocNext`	A parameter value for `LDC` and `STC` callback functions to indicate that this is a subsequent data transfer in a sequence.
`0x12`	`CoprocNop`	Treat as a NOP.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

2.6.2 CounterInterface protocol

This protocol connects the `cntvalueb` port on Generic Timer components to MemoryMappedCounterModule components.

This semi-opaque protocol is exportable across a SystemC interface using a custom bridge.

It is a LISA+ protocol.

2.6.3 GICv3Comms protocol

This protocol connects the `gicv3_redistributor_s` port on components with GICv3 Core Interfaces to platform level GICv3 components.

This semi-opaque protocol is exportable across a SystemC interface using the bridges GICv3CommsPVBus and PVBusGICv3Comms.

It is a LISA+ protocol.

2.6.4 InstructionCount protocol

This protocol has the behaviors `getValue()` and `getRunState()`.

`getValue() : uint64_t`
Obtain the number of instructions executed by the processor.

`getRunState() : uint32_t`
Obtain the power/run status of the processor.

Table 2-6 Run state values

Value	State label	Description
`0x0`	`UNKNOWN`	Run status unknown, that is, simulation has not started
`0x1`	`RUNNING`	Processor running, is not idle and is executing instructions
`0x2`	`HALTED`	External halt signal asserted
`0x3`	`STANDBY_WFE`	Last instruction executed was WFE and standby mode has been entered
`0x4`	`STANDBY_WFI`	Last instruction executed was WFI and standby mode has been entered
`0x5`	`IN_RESET`	External reset signal asserted
`0x6`	`DORMANT`	Partial processor power down
`0x7`	`SHUTDOWN`	Complete processor power down

2.6.5 v8EmbeddedCrossTrigger_controlprotocol protocol

This protocol connects the *Cross Trigger Interface* (CTI) on ARMv8 processor components to platform level *Cross Trigger Matrix* (CTM) components.

This opaque protocol is not exportable across a SystemC interface.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

2.7 Signaling protocols

This section describes the signaling protocols.

This section contains the following subsections:

- *2.7.1 Signaling protocols - about* on page 2-85.
- *2.7.2 Signal protocol* on page 2-85.
- *2.7.3 StateSignal protocol* on page 2-85.
- *2.7.4 Value protocol* on page 2-85.
- *2.7.5 Value_64 protocol* on page 2-85.
- *2.7.6 ValueState protocol* on page 2-85.

2.7.1 Signaling protocols - about

Many components use the signaling protocols to indicate changes in state for signals such as interrupt and reset.

Each signaling protocol has two variants:

- One permits components to signal a state change to other components.
- One permits the other components to passively query the current state of the signal.

2.7.2 Signal protocol

The Signal protocol has the setValue behavior.

`setValue(enum sg::Signal::State) : void`
indicates a state change.

Legal values for `sg::Signal::State` are:

- `sg::Signal::Set`.
- `sg::Signal::Clear`.

2.7.3 StateSignal protocol

The StateSignal protocol has the setValue and getValue behaviors.

`setValue(enum sg::Signal::State) : void`
indicates a value change.

`getValue() : sg::Signal::State`
reads the current value.

2.7.4 Value protocol

The Value protocol has the setValue behavior.

`setValue(uint32_t value)`
indicates a state change.

2.7.5 Value_64 protocol

This protocol provides the rules to communicate with the TrustZone® Memory Adapter (TZMA) to signal the remapped range X.

`setValue(uint64_t value)`
Sets a 64-bit wide address to the slave port.

2.7.6 ValueState protocol

The ValueState protocol has the setValue and getValue behaviors.

`setValue(uint32_t value) : void`
indicates a value change.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

`getValue() : uint32_t`
reads the current value state.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Chapter 3
Fast Models components

This chapter describes all model components in Fast Models, organized by component type.

It contains the following sections:

- *3.1 Bridge components* on page 3-88.
- *3.2 Bus components* on page 3-102.
- *3.3 Clocking components* on page 3-113.
- *3.4 Core components* on page 3-117.
- *3.5 Media components* on page 3-601.
- *3.6 Other components* on page 3-616.
- *3.7 Peripheral components* on page 3-618.
- *3.8 Signals components* on page 3-644.
- *3.9 SystemIP components* on page 3-647.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

3.1 Bridge components

This section describes the Bridge components.

These components allow conversion between the following protocols:

- PVBus and AMBAPV.
- Signal and AMBAPVSignal.
- StateSignal and AMBAPVSignalState.
- Value(_64) and AMBAPVValue(64).
- ValueState(_64) and AMBAPVValueState(64).

LISA+ source for the bridge components is located in `$PVLIB_HOME/examples/SystemCExport/Bridges/`.

The AMBAPV protocols and components are designed to interface with the AMBA TLM PV library for ASI TLM 2.0. Fast Models provides this library as a standard way of mapping the AMBA protocol on top of ASI TLM 2.0.2 kit at PV level.

For more information about the AMBA TLM PV library for ASI TLM 2.0.2 kit, see the Fast Models documentation in `%MAXCORE_HOME%\AMBA-PV\doc`. On Linux, use the `$MAXCORE_HOME` environment variable instead.

For more information about ASI TLM 2.0, see the Accellera documentation that is provided with the kit.

This section contains the following subsections:

- *3.1.1 AMBAPV2PVBus* on page 3-89.
- *3.1.2 AMBAPVACE2PVBus* on page 3-89.
- *3.1.3 AMBAPVSignal2SGSignal* on page 3-89.
- *3.1.4 AMBAPVSignalState2SGStateSignal* on page 3-90.
- *3.1.5 AMBAPVValue2SGValue* on page 3-90.
- *3.1.6 AMBAPVValue2SGValue64* on page 3-90.
- *3.1.7 AMBAPVValue642SMMUv3AEMIdentify* on page 3-91.
- *3.1.8 AMBAPVValueState2SGValueState* on page 3-91.
- *3.1.9 AMBAPVValueState2SGValueState64* on page 3-91.
- *3.1.10 BroadcastSignal2AMBAPVSignal* on page 3-91.
- *3.1.11 Clock2SystemC* on page 3-92.
- *3.1.12 CounterInterface2SystemC* on page 3-92.
- *3.1.13 InstructionCount2SystemC* on page 3-92.
- *3.1.14 LCD2SystemC* on page 3-93.
- *3.1.15 PChannel2SystemC* on page 3-93.
- *3.1.16 PVBus2AMBAPV* on page 3-94.
- *3.1.17 PVBus2AMBAPVACE* on page 3-95.
- *3.1.18 PVBusBridge* on page 3-96.
- *3.1.19 SGSignal2AMBAPVSignal* on page 3-96.
- *3.1.20 SGStateSignal2AMBAPVSignalState* on page 3-96.
- *3.1.21 SGValue2AMBAPVValue* on page 3-97.
- *3.1.22 SGValue2AMBAPVValue64* on page 3-97.
- *3.1.23 SGValueState2AMBAPVValueState* on page 3-97.
- *3.1.24 SGValueState2AMBAPVValueState64* on page 3-98.
- *3.1.25 SMMUv3AEMIdentify2AMBAPVValue64* on page 3-98.
- *3.1.26 SystemC2Clock* on page 3-98.
- *3.1.27 SystemC2InstructionCount* on page 3-99.
- *3.1.28 SystemC2LCD* on page 3-99.
- *3.1.29 SystemC2PChannel* on page 3-99.
- *3.1.30 SystemC2VirtualEthernet* on page 3-100.
- *3.1.31 SystemC2v7VGICConfig* on page 3-100.
- *3.1.32 VirtualEthernet2SystemC* on page 3-100.
- *3.1.33 v7VGICConfig2SystemC* on page 3-101.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

3.1.1 AMBAPV2PVBus

AMBA-PV to PVBus protocol converter. This model is written in LISA+.

AMBAPV2PVBus contains the following CADI targets:

- AMBAPV2PVBus

AMBAPV2PVBus contains the following MTI components:

- *4.106 PVBusMaster - trace* on page 4-2759

About AMBAPV2PVBus

- PVBus does not support transactions with `byte_enable` set (strobing transactions, in AXI terms). This bridge component rejects them.
- Variants of this component also exist with multiple input and output ports.

Table 3-1 Ports

Name	Protocol	Type	Description
amba_pv_s	*AMBAPV* on page 2-59	Slave	-
pvbus_m	PVBus	Master	-

Table 3-2 Parameters for AMBAPV2PVBus

Name	Type	Default value	Description
base_addr	int	0x0	Base address
shareable	bool	0x1	Shareable default

3.1.2 AMBAPVACE2PVBus

AMBA-PV ACE to PVBus protocol converter. This model is written in LISA+.

AMBAPVACE2PVBus contains the following CADI targets:

- AMBAPVACE2PVBus

AMBAPVACE2PVBus contains the following MTI components:

- *4.106 PVBusMaster - trace* on page 4-2759

About AMBAPVACE2PVBus

- AMBAPVACE2PVBus depends on the AMBA-PV API, which must be at least version 1.4.
- The translation of bus transactions by the bridge has some impact on performance. Bus masters that cache memory transactions avoid much of this impact. The bridge does not support DMI.

Table 3-3 Ports

Name	Protocol	Type	Description
amba_pv_ace_s	*AMBAPVACE* on page 2-61	Slave	-
pvbus_m	PVBus	Master	-

3.1.3 AMBAPVSignal2SGSignal

AMBA-PV Signal to SystemGenerator Signal protocol converter. This model is written in LISA+.

Note

Variants of this component also exist with multiple input and output ports.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-4 Ports

Name	Protocol	Type	Description
amba_pv_signal_s	*AMBAPVSignal* on page 2-63	Slave	Input slave port for connection from top-level AMBAPVSignal slave port.
sg_signal_m	*Signal* on page 2-85	Master	Handles outgoing signal state changes. Converted signal state changes are sent out through this port.

3.1.4 AMBAPVSignalState2SGStateSignal

AMBA-PV SignalState to SystemGenerator StateSignal protocol converter. This model is written in LISA+.

Note

Variants of this component also exist with multiple input and output ports.

Table 3-5 Ports

Name	Protocol	Type	Description
amba_pv_signal_s	*AMBAPVSignalState* on page 2-63	Slave	-
sg_signal_m	*StateSignal* on page 2-85	Master	-

3.1.5 AMBAPVValue2SGValue

AMBA-PV Value to SystemGenerator Value protocol converter. This model is written in LISA+.

Note

Variants of this component also exist with multiple input and output ports.

Table 3-6 Ports

Name	Protocol	Type	Description
amba_pv_value_s	*AMBAPVValue* on page 2-63	Slave	-
sg_value_m	*Value* on page 2-85	Master	-

3.1.6 AMBAPVValue2SGValue64

AMBA-PV Value64 to SystemGenerator Value_64 protocol converter. This model is written in LISA+.

Note

Variants of this component also exist with multiple input and output ports.

Table 3-7 Ports

Name	Protocol	Type	Description
amba_pv_value_s	*AMBAPVValue64* on page 2-63	Slave	-
sg_value_m	*Value_64* on page 2-85	Master	-

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

3.1.7 AMBAPVValue642SMMUv3AEMIdentify

AMBA-PV Value64 to SMMUv3AEMIdentify protocol converter. This model is written in LISA+.

Table 3-8 Ports

Name	Protocol	Type	Description
identify	*SMMUv3AEMIdentifyProtocol* on page 2-78	Master	-
identify_reply	*AMBAPVValue64* on page 2-63	Master	-
identify_request	*AMBAPVValue64* on page 2-63	Slave	-

3.1.8 AMBAPVValueState2SGValueState

AMBA-PV ValueState to SystemGenerator ValueState protocol converter. This model is written in LISA+.

Note

Variants of this component also exist with multiple input and output ports.

Table 3-9 Ports

Name	Protocol	Type	Description
amba_pv_value_s	*AMBAPVValueState* on page 2-64	Slave	-
sg_value_m	*ValueState* on page 2-85	Master	-

3.1.9 AMBAPVValueState2SGValueState64

AMBA-PV ValueState64 to SystemGenerator ValueState_64 protocol converter. This model is written in LISA+.

Note

Variants of this component also exist with multiple input and output ports.

Table 3-10 Ports

Name	Protocol	Type	Description
amba_pv_value_s	*AMBAPVValueState64* on page 2-64	Slave	-
sg_value_m	ValueState_64	Master	-

3.1.10 BroadcastSignal2AMBAPVSignal

Broadcast signal to AMBAPVSignal coverter. This model is written in LISA+.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

BroadcastSignal2AMBAPVSignal contains the following CADI targets:

- BroadcastSignal2AMBAPVSignal

Table 3-11 Ports

Name	Protocol	Type	Description
amba_pv_signal_m	*AMBAPVSignal* on page 2-63	Master	-
amba_pv_signal_s	*AMBAPVSignal* on page 2-63	Slave	-
b_signal	*Signal* on page 2-85	Broadcast	-

3.1.11 Clock2SystemC

Clock to SystemC Converter. This model is written in LISA+.

Clock2SystemC contains the following CADI targets:

- Clock2SystemC

Table 3-12 Ports

Name	Protocol	Type	Description
clk_in	ClockSignal	Slave	-
current_ticks_s	*AMBAPVValueState64* on page 2-64	Slave	-
get_clock_s	*AMBAPVValueState64* on page 2-64	Slave	-
rate_in_Hz_s	*AMBAPVValueState64* on page 2-64	Slave	-
set_clock_m	*AMBAPVValue64* on page 2-63	Master	-

3.1.12 CounterInterface2SystemC

CounterInterface to SystemC Converter. This model is written in LISA+.

CounterInterface2SystemC contains the following CADI targets:

- CounterInterface2SystemC

Table 3-13 Ports

Name	Protocol	Type	Description
amba_pv_eventUpdate_m	*AMBAPVValue* on page 2-63	Master	-
amba_pv_getCounterValue_s	*AMBAPVValueState64* on page 2-64	Slave	-
amba_pv_requestEventUpdate_s	*AMBAPVValue64* on page 2-63	Slave	-
amba_pv_requestSignalUpdate_s	*AMBAPVValue64* on page 2-63	Slave	-
amba_pv_setEnabled_m	*AMBAPVValue* on page 2-63	Master	-
amba_pv_signalUpdate_m	*AMBAPVValue* on page 2-63	Master	-
cntvalueb	*CounterInterface* on page 2-84	Slave	-

3.1.13 InstructionCount2SystemC

InstructionCount to SystemC Converter. This model is written in LISA+.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

InstructionCount2SystemC contains the following CADI targets:

- InstructionCount2SystemC

Note

Variants of this component also exist with multiple input and output ports.

Table 3-14 Ports

Name	Protocol	Type	Description
inst_count	*AMBAPVValueState64* on page 2-64	Slave	-
run_state	*AMBAPVValueState* on page 2-64	Slave	-
ticks	*InstructionCount* on page 2-84	Slave	-

3.1.14 LCD2SystemC

Converts LCD protocol to SystemC. This model is written in LISA+.

LCD2SystemC contains the following CADI targets:

- LCD2SystemC

Table 3-15 Ports

Name	Protocol	Type	Description
all_received_sPL	*AMBAPVSignal* on page 2-63	Master	-
all_received_u	*AMBAPVSignal* on page 2-63	Master	-
lcd_s	*LCD* on page 2-72	Slave	-
lock_m	*AMBAPVValueState64* on page 2-64	Master	-
setPreferredLayout_d	*AMBAPVValue* on page 2-63	Master	-
setPreferredLayout_h	*AMBAPVValue* on page 2-63	Master	-
setPreferredLayout_w	*AMBAPVValue* on page 2-63	Master	-
unlock_m	*AMBAPVSignal* on page 2-63	Master	-
update_h	*AMBAPVValue* on page 2-63	Master	-
update_w	*AMBAPVValue* on page 2-63	Master	-
update_x	*AMBAPVValue* on page 2-63	Master	-
update_y	*AMBAPVValue* on page 2-63	Master	-

3.1.15 PChannel2SystemC

PChannel to SystemC Converter. This model is written in LISA+.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PChannel2SystemC contains the following CADI targets:

- PChannel2SystemC

Table 3-16 Ports

Name	Protocol	Type	Description
pchannel	*PChannel* on page 2-80	Slave	-
sc_pchannel	SystemCPChannel	Master	-

3.1.16 PVBus2AMBAPV

PVBus to AMBA-PV protocol converter. This model is written in LISA+.

PVBus2AMBAPV contains the following CADI targets:

- PVBus2AMBAPV

PVBus2AMBAPV contains the following MTI components:

- *4.101 PVBusBridge - trace* on page 4-2751

About PVBus2AMBAPV

DMI viewer provides the debugging functionality of the PVBus2AMBAPV bridge. When activated, it dumps the content of the DMI cache in the bridge in the following CSV format:

```
Range start, Range end incl, Pointer, Latency, R/W, Attributes
```

To be able to activate this functionality, a name for the counters output file must be set, using the `counters-file-name` parameter. If the counters file name is set, when `dump-dmi-cache` is set to 1 at runtime, the DMI cache of the bridge is dumped. The runtime parameter is always reset to 0 when the dump has completed.

Note

Variants of this component also exist with multiple input and output ports.

Table 3-17 Ports

Name	Protocol	Type	Description
amba_pv_m	*AMBAPV* on page 2-59	Master	-
pvbus_s	PVBus	Slave	-

Table 3-18 Parameters for PVBus2AMBAPV

Name	Type	Default value	Description
counters-file-name	string	""	Prefix of the file name to store counters at the end of simulation
dmi-container-type	string	"TZAttr"	Type of the DMI cache. Allowed values: TZAttr and FullAttr
dump-dmi-cache	bool	0x0	Dumps the content of the DMI cache into a file
dump-dmi-file-name	string	""	Prefix of the file name to dump the content of the DMI when requested
force-dmi-size	bool	0x1	Force DMI start and end address to be 4kB-aligned
min-range-to-cache	int	0x10000	Min DMI range size to cache in the bridge
size	int	0x1000000000000	Maximum size of memory region

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

3.1.17 PVBus2AMBAPVACE

PVBus to AMBA-PV ACE protocol converter. This model is written in LISA+.

PVBus2AMBAPVACE contains the following CADI targets:

- PVBus2AMBAPVACE

PVBus2AMBAPVACE contains the following MTI components:

- *4.100 PVBus2AMBAPVACE - trace* on page 4-2750
- *4.101 PVBusBridge - trace* on page 4-2751
- *4.105 PVBusMapper - trace* on page 4-2758

About PVBus2AMBAPVACE

PVBus2AMBAPVACE depends on the AMBA-PV API, which must be at least version 1.4.

The translation of bus transactions by the bridge has some impact on performance. Bus masters that cache memory transactions avoid much of this impact.

DMI viewer provides the debugging functionality of the PVBus2AMBAPV bridge. When activated, it dumps the content of the DMI cache in the bridge in the following CSV format:

```
Range start, Range end incl, Pointer, Latency, R/W, Attributes
```

To be able to activate this functionality, a name for the counters output file must be set, using the `counters-file-name` parameter. If the counters file name is set, when `dump-dmi-cache` is set to 1 at runtime, the DMI cache of the bridge is dumped. The runtime parameter is always reset to 0 when the dump has completed.

Table 3-19 Ports

Name	Protocol	Type	Description
amba_pv_ace_m	*AMBAPVACE on page 2-61*	Master	-
pvbus_over_tlm_control	PVBusOverTLMControl	Slave	-
pvbus_s	PVBus	Slave	-

Table 3-20 Parameters for PVBus2AMBAPVACE

Name	Type	Default value	Description
counters-file-name	string	""	Prefix of the file name to store counters at the end of simulation
dmi-cache-name	string	""	DEPRECATED: This parameter will be ignored. Name of the DMI cache. Useful for multiple bridges to share the same cache
dmi-container-type	string	"TZAttr"	Type of the DMI cache. Allowed values: TZAttr and FullAttr. When there are caches downstream of this bridge and cache-state-modelled=1 and route-tlm=1 use only FullAttr.
dump-dmi-cache	bool	0x0	Dumps the content of the DMI cache into a file
dump-dmi-file-name	string	""	Prefix of the file name to dump the content of the DMI when requested
force-dmi-size	bool	0x1	Force DMI start and end address to be 4kB-aligned
min-range-to-cache	int	0x10000	Min DMI range size to cache in the bridge

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-20 Parameters for PVBus2AMBAPVACE (continued)

Name	Type	Default value	Description
route-tlm	bool	0x1	Route all the PVBus traffic explicitly to the TLM bus. Allows to monitor transactions on the TLM bus but slows down the emulation. The routing must always be to TLM if there is not a corresponding AMBAPVACE2PVBus bridge downstream.
set-ace-lite	bool	0x0	Set bridge mode when connecting to ace-lite ports. If true, the bridge will not deal with SNOOPs.
size	int	0x1000000000000	Maximum size of memory region, i.e. the first unsupported address.

3.1.18 PVBusBridge

A PVBusBridge bridges incoming transactions to a PVDevice port. This model is written in C++.

PVBusBridge contains the following MTI components:

- *4.101 PVBusBridge - trace* on page 4-2751

Table 3-21 Ports

Name	Protocol	Type	Description
control	PVBusBridgeControl	Slave	Control signal.
device	*PVDevice* on page 2-77	Master	Optimised connection out to devices.
dump_dmi	*Signal* on page 2-85	Slave	On the assert of this signal the bridge will dump dmi cache content into a csv file
pvbus_s	PVBus	Slave	Connection in from bus master.
reset	*Signal* on page 2-85	Slave	On the assert of this signal, a reset of the bus slave will be latched this is used by the bus deadlock detection logic.

3.1.19 SGSignal2AMBAPVSignal

SystemGenerator Signal to AMBA-PV Signal protocol converter. This model is written in LISA+.

Note

Variants of this component also exist with multiple input and output ports.

Table 3-22 Ports

Name	Protocol	Type	Description
amba_pv_signal_m	*AMBAPVSignal* on page 2-63	Master	Output master port for connection to top-level AMBAPVSignal master port. Converted signal state changes are sent out through this port.
sg_signal_s	*Signal* on page 2-85	Slave	Handles incoming signal state changes.

3.1.20 SGStateSignal2AMBAPVSignalState

SystemGenerator StateSignal to AMBA-PV SignalState protocol converter. This model is written in LISA+.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Note

Variants of this component also exist with multiple input and output ports.

Table 3-23 Ports

Name	Protocol	Type	Description
amba_pv_signal_m	*AMBAPVSignalState* on page 2-63	Master	Output master port for connection to top-level AMBAPVValue64 master port. Converted value changes are sent out through this port.
sg_signal_s	*StateSignal* on page 2-85	Slave	Handles incoming value changes.

3.1.21 SGValue2AMBAPVValue

SystemGenerator Value to AMBA-PV Value protocol converter. This model is written in LISA+.

Note

Variants of this component also exist with multiple input and output ports.

Table 3-24 Ports

Name	Protocol	Type	Description
amba_pv_value_m	*AMBAPVValue* on page 2-63	Master	Output master port for connection to top-level AMBAPVValue master port. Converted value changes are sent out through this port.
sg_value_s	*Value* on page 2-85	Slave	Handles incoming value changes.

3.1.22 SGValue2AMBAPVValue64

SystemGenerator Value_64 to AMBA-PV Value64 protocol converter. This model is written in LISA+.

Note

Variants of this component also exist with multiple input and output ports.

Table 3-25 Ports

Name	Protocol	Type	Description
amba_pv_value_m	*AMBAPVValue64* on page 2-63	Master	Output master port for connection to top-level AMBAPVValue master port. Converted value changes are sent out through this port.
sg_value_s	*Value_64* on page 2-85	Slave	Handles incoming value changes.

3.1.23 SGValueState2AMBAPVValueState

SystemGenerator ValueState to AMBA-PV ValueState protocol converter. This model is written in LISA+.

Note

Variants of this component also exist with multiple input and output ports.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-26 Ports

Name	Protocol	Type	Description
amba_pv_value_m	*AMBAPVValueState on page 2-64*	Master	Output master port for connection to top-level AMBAPVValue master port. Converted value changes are sent out through this port.
sg_value_s	*ValueState on page 2-85*	Slave	Handles incoming value changes.

3.1.24 SGValueState2AMBAPVValueState64

SystemGenerator ValueState_64 to AMBA-PV ValueState64 protocol converter. This model is written in LISA+.

Note

Variants of this component also exist with multiple input and output ports.

Table 3-27 Ports

Name	Protocol	Type	Description
amba_pv_value_m	*AMBAPVValueState64 on page 2-64*	Master	Output master port for connection to top-level AMBAPVValue64 master port. Converted value changes are sent out through this port.
sg_value_s	ValueState_64	Slave	Handles incoming value changes.

3.1.25 SMMUv3AEMIdentify2AMBAPVValue64

SMMUv3AEMIdentify to AMBA-PV Value64 protocol converter. This model is written in LISA+.

Table 3-28 Ports

Name	Protocol	Type	Description
identify	*SMMUv3AEMIdentifyProtocol on page 2-78*	Slave	SMMUv3AEMIdentifyProtocol input.
identify_reply	*AMBAPVValue64 on page 2-63*	Slave	From SystemC.
identify_request	*AMBAPVValue64 on page 2-63*	Master	To SystemC.

3.1.26 SystemC2Clock

Clock to SystemC Converter. This model is written in LISA+.

SystemC2Clock contains the following CADI targets:

- SystemC2Clock

Table 3-29 Ports

Name	Protocol	Type	Description
clk_out	ClockSignal	Master	ClockSignal ouput
current_ticks_m	*AMBAPVValueState64 on page 2-64*	Master	To SystemC.
get_clock_m	*AMBAPVValueState64 on page 2-64*	Master	To SystemC.
rate_in_Hz_m	*AMBAPVValueState64 on page 2-64*	Master	To SystemC.
set_clock_s	*AMBAPVValue64 on page 2-63*	Slave	From SystemC.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

3.1.27 SystemC2InstructionCount

SystemC to InstructionCount Converter. This model is written in LISA+.

SystemC2InstructionCount contains the following CADI targets:

- SystemC2InstructionCount

Note

Variants of this component also exist with multiple input and output ports.

Table 3-30 Ports

Name	Protocol	Type	Description
inst_count	*AMBAPVValueState64* on page 2-64	Master	To SystemC to request instruction count.
run_state	*AMBAPVValueState* on page 2-64	Master	To SystemC to request run state.
ticks	*InstructionCount* on page 2-84	Master	InstructionCount input.

3.1.28 SystemC2LCD

Converts SystemC to LCD protocol. This model is written in LISA+.

SystemC2LCD contains the following CADI targets:

- SystemC2LCD

Table 3-31 Ports

Name	Protocol	Type	Description
all_received_sPL	*AMBAPVSignal* on page 2-63	Slave	From SystemC.
all_received_u	*AMBAPVSignal* on page 2-63	Slave	From SystemC.
lcd_m	*LCD* on page 2-72	Master	LCD output.
lock_s	*AMBAPVValueState64* on page 2-64	Slave	From SystemC.
setPreferredLayout_d	*AMBAPVValue* on page 2-63	Slave	From SystemC.
setPreferredLayout_h	*AMBAPVValue* on page 2-63	Slave	From SystemC.
setPreferredLayout_w	*AMBAPVValue* on page 2-63	Slave	From SystemC.
unlock_s	*AMBAPVSignal* on page 2-63	Slave	From SystemC.
update_h	*AMBAPVValue* on page 2-63	Slave	From SystemC.
update_w	*AMBAPVValue* on page 2-63	Slave	From SystemC.
update_x	*AMBAPVValue* on page 2-63	Slave	From SystemC.
update_y	*AMBAPVValue* on page 2-63	Slave	From SystemC.

3.1.29 SystemC2PChannel

SystemC to PChannel Converter. This model is written in LISA+.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SystemC2PChannel contains the following CADI targets:

- SystemC2PChannel

Table 3-32 Ports

Name	Protocol	Type	Description
pchannel	*PChannel* on page 2-80	Master	-
sc_pchannel	SystemCPChannel	Slave	-

3.1.30 SystemC2VirtualEthernet

SystemC to VirtualEthernet Converter. This model is written in LISA+.

SystemC2VirtualEthernet contains the following CADI targets:

- SystemC2VirtualEthernet

Table 3-33 Ports

Name	Protocol	Type	Description
virtualethernet_m	*VirtualEthernet* on page 2-79	Master	-
virtualethernet_s	SC_VirtualEthernet	Slave	-

3.1.31 SystemC2v7VGICConfig

Converts SystemC to v7_vgic_configuration_protocol. This model is written in LISA+.

SystemC2v7VGICConfig contains the following CADI targets:

- SystemC2v7VGICConfig

Table 3-34 Ports

Name	Protocol	Type	Description
all_received_s	*AMBAPVSignal* on page 2-63	Slave	From SystemC.
cpu_interface_number_s	*AMBAPVValue64* on page 2-63	Slave	From SystemC.
inout_cluster_number_s	*AMBAPVValue64* on page 2-63	Slave	From SystemC.
inout_cpu_number_in_cluster_s	*AMBAPVValue64* on page 2-63	Slave	From SystemC.
master_id_mask_s	*AMBAPVValue* on page 2-63	Slave	From SystemC.
master_id_s	*AMBAPVValue* on page 2-63	Slave	From SystemC.
number_of_cores_s	*AMBAPVValueState* on page 2-64	Slave	From SystemC.
response_m	*AMBAPVSignal* on page 2-63	Master	To SystemC.
v7_vgic_config_m	v7_VGIC_Configuration_Protocol	Master	v7_VGIC_Configuration_Protocol output.

3.1.32 VirtualEthernet2SystemC

VirtualEthernet to SystemC Converter. This model is written in LISA+.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VirtualEthernet2SystemC contains the following CADI targets:
- VirtualEthernet2SystemC

Table 3-35 Ports

Name	Protocol	Type	Description
virtualethernet_m	SC_VirtualEthernet	Master	-
virtualethernet_s	*VirtualEthernet* on page 2-79	Slave	-

3.1.33 v7VGICConfig2SystemC

Converts v7_vgic_configuration_protocol to SystemC. This model is written in LISA+.

v7VGICConfig2SystemC contains the following CADI targets:
- v7VGICConfig2SystemC

Table 3-36 Ports

Name	Protocol	Type	Description
all_received	*AMBAPVSignal* on page 2-63	Master	Called when all other values have been set in opposite bridge.
cpu_interface_number_m	*AMBAPVValue64* on page 2-63	Master	To SystemC.
inout_cluster_number_m	*AMBAPVValue64* on page 2-63	Master	To SystemC.
inout_cpu_number_in_cluster_m	*AMBAPVValue64* on page 2-63	Master	To SystemC.
master_id_m	*AMBAPVValue* on page 2-63	Master	To SystemC.
master_id_mask_m	*AMBAPVValue* on page 2-63	Master	To SystemC.
number_of_cores_m	*AMBAPVValueState* on page 2-64	Master	To SystemC.
response_s	*AMBAPVSignal* on page 2-63	Slave	From SystemC.
v7_vgic_config_s	v7_VGIC_Configuration_Protocol	Slave	v7_VGIC_Configuration_Protocol input.

Related concepts

2.1.1 AMBA-PV protocols - about on page 2-59

Related information

AMBA-PV Extensions to TLM 2.0 Developer Guide

Fast Models User Guide, SystemC Export with Multiple Instantiation

Accellera Systems Initiative

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

3.2 Bus components

This section describes the Bus components.

PVBus components and protocols provide functionally-accurate communication between bus masters and slaves. They are not software implementations of specific hardware, but instead are abstract components that are required by the software model environment.

There is no modeling of handshaking or cycle counts. By removing this level of detail, and by using efficient internal communication mechanisms, PVBus components can provide very fast modeling of bus accesses. A number of components, which abstract the internal details, access the PVBus API. You must use these components correctly to achieve high simulation speeds.

This section contains the following subsections:

- *3.2.1 Labeller* on page 3-102.
- *3.2.2 LabellerForDMA330* on page 3-103.
- *3.2.3 LabellerForGPUProtMode* on page 3-103.
- *3.2.4 MSIRewriter* on page 3-104.
- *3.2.5 PVBusCache* on page 3-105.
- *3.2.6 PVBusDecoder* on page 3-106.
- *3.2.7 PVBusExclusiveMonitor* on page 3-106.
- *3.2.8 PVBusExclusiveSquasher* on page 3-107.
- *3.2.9 PVBusLogger* on page 3-107.
- *3.2.10 PVBusMapper* on page 3-108.
- *3.2.11 PVBusMaster* on page 3-108.
- *3.2.12 PVBusModifier* on page 3-109.
- *3.2.13 PVBusRouter* on page 3-109.
- *3.2.14 PVBusSlave* on page 3-109.
- *3.2.15 PVWriteBuffer* on page 3-110.
- *3.2.16 SimplePVBusMaster* on page 3-111.
- *3.2.17 TZSwitch* on page 3-111.

3.2.1 Labeller

This model is written in LISA+.

Labeller contains the following CADI targets:

- Labeller

Labeller contains the following MTI components:

- *4.105 PVBusMapper - trace* on page 4-2758

About Labeller

Labeller and LabellerForDMA330 are utility components that allow the system designer to embed values into the Label field for transactions generated by a Bus Master. They go between PVBus Master and Slave ports.

The following example creates a labeller to add an ID for an HDLCD controller that is upstream of a TZC_400. The system designer specifies a unique set of IDs for use as *Non-Secure Access IDs* (NSAIDs) in the TZC_400. The labeller can insert these IDs directly into the transaction.

```
pl370_hdlcd : PL370_HDLCD();
hdlcd_labeller : Labeller( "label" = 2 );
pl370_hdlcd.pvbus_m => hdlcd_labeller.pvbus_s;
hdlcd_labeller.pvbus_m => output_bus.pvbus_s;
```


Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-37 Ports

Name	Protocol	Type	Description
pvbus_m	PVBus	Master	Output with modified MasterID.
pvbus_s	PVBus	Slave	Unmodified input.

Table 3-38 Parameters for Labeller

Name	Type	Default value	Description
label	int	0x0	The label to apply to all transactions flowing through the labeller

3.2.2 LabellerForDMA330

This model is written in LISA+.

LabellerForDMA330 contains the following CADI targets:

- LabellerForDMA330

LabellerForDMA330 contains the following MTI components:

- *4.105 PVBusMapper - trace* on page 4-2758

About LabellerForDMA330

Labeller and LabellerForDMA330 are utility components that allow the system designer to embed values into the Label field for transactions generated by a Bus Master. They go between PVBus Master and Slave ports.

Table 3-39 Ports

Name	Protocol	Type	Description
pvbus_m	PVBus	Master	Output with modified MasterID.
pvbus_s	PVBus	Slave	Unmodified input.

Table 3-40 Parameters for LabellerForDMA330

Name	Type	Default value	Description
dma330_data_label	int	0x0	The label to apply to all _data_ transactions flowing through the labeller. Used as a base value in conjunction with the channel ID if data-channel discrimination is enabled.
dma330_discriminate_data_channels	bool	0x0	Discriminate between DMA-330 data channels. Channel ID is added to the data label.
dma330_ns_instruction_label	int	0x0	The label to apply to all non-secure _instructions_ transactions flowing through the labeller
dma330_s_instruction_label	int	0x0	The label to apply to all secure _instructions_ transactions flowing through the labeller

3.2.3 LabellerForGPUProtMode

This model is written in LISA+.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LabellerForGPUProtMode contains the following CADI targets:

- LabellerForGPUProtMode

LabellerForGPUProtMode contains the following MTI components:

- *4.105 PVBusMapper - trace* on page 4-2758

About LabellerForGPUProtMode

This component adds Non-Secure Access IDs (NSAIDs) to the transactions generated by the GPU. The NSAID is a four-bit number. It allows other components, such as a TrustZone Controller (TZC) or a Dynamic Memory Controller (DMC) to filter transactions and control access to memory regions that are designated as protected.

Table 3-41 Ports

Name	Protocol	Type	Description
prot_mode	*Signal* on page 2-85	Slave	Input to determine whether output is supposed to be protected or not
pvbus_m	PVBus	Master	Output with modified MasterID.
pvbus_s	PVBus	Slave	Unmodified input.

Table 3-42 Parameters for LabellerForGPUProtMode

Name	Type	Default value	Description
gpu_id_normal	int	0x0	NSAID to apply to all transactions flowing through the labeller when prot_mode is low.
gpu_id_protected	int	0x0	NSAID to apply to all transactions flowing through the labeller when prot_mode is high.

3.2.4 MSIRewriter

Recognise writes to the GITS_TRANSLATER register and rewrite them to go to the GITS_TRANSLATE64R register. The DeviceID is expected to be in the bottom 32 bits of ExtendedID of the transaction. Debug transactions and reads are not rewritten. This component can also, optionally, apply a 16 bit 'label' to the top 16 bits of the MasterID in the same way that the Labeller component does. This model is written in LISA+.

MSIRewriter contains the following CADI targets:

- MSIRewriter

MSIRewriter contains the following MTI components:

- *4.86 MSIRewriter - trace* on page 4-2723
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.106 PVBusMaster - trace* on page 4-2759
- *4.107 PVBusSlave - trace* on page 4-2760

MasterID, ExtendedID, and UserFlags

These tables show how this model encodes information in the `MasterID`, `ExtendedID`, and `UserFlags` bus attributes:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-43 pvbus_s port

PVBus attribute	Bits used	Property encoded	Notes
MasterID	-	-	Not used.
ExtendedID	Bits[63:32]	Stream ID	-
	Bits[31:0]	Device ID	-
UserFlags	-	-	Not used.

Table 3-44 pvbus_m port

PVBus attribute	Bits used	Property encoded	Notes
MasterID	Bits[31:16]	Optional 16-bit label.	For usage, see the label parameter.
ExtendedID	-	-	Not used.
UserFlags	-	-	Not used.

Table 3-45 Ports

Name	Protocol	Type	Description
pvbus_m	PVBus	Master	-
pvbus_s	PVBus	Slave	-

Table 3-46 Parameters for MSIRewriter

Name	Type	Default value	Description
GITS_TRANSLATE64R_OFFSET	int	0x10048	It is an offset from ITS0-Base.
ITS0-base	int	0x0	Register base address for ITS0. This base address is used to recognise writes to the GITS_TRANSLATER register within the ITS0's register frame.
enable_rewriting	bool	0x1	Enable rewriting.
label	int	0xffffffff	If < 2**16 then this is a label that is put in the top 16 bits of MasterID in the same way that the component Labeller does. This labelling is not controlled by enable_rewriting and is performed on all transactions (even rewritten ones)
log	int	0x0	Log level, 0 is off.

3.2.5 PVBusCache

A PVBusCache manages cache-line data and supports forwarding of transactions. This model is written in C++.

Table 3-47 Ports

Name	Protocol	Type	Description
bus_in[4]	PVBus	Slave	Connections in from bus master.
bus_out[4]	PVBus	Master	Connections out to bus slaves.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-47 Ports (continued)

Name	Protocol	Type	Description
control	PVBusCacheControl	Slave	Configuration/control port.
device	PVBusCacheDevice	Master	Connection out to cache device.

3.2.6 PVBusDecoder

A PVBusDecoder allows bus transactions to be routed to one of many slaves, based on the address given in the transaction. This model is written in C++.

About PVBusDecoder

Each slave connection is associated with a specific address range on the pvbus_m_range port. In LISA+, the syntax for this is:

```
decoder.pvbus_m_range[start..end] = slave.pvbus
```

The values for start (inclusive) and end (inclusive) must specify a 4KB-aligned region of a multiple of 4K bytes. You can specify an address range for the slave, where the decoder remaps addresses into the appropriate range. The default address range for a slave is [0-(*sizeOfMasterRange* – 1)].

Table 3-48 Ports

Name	Protocol	Type	Description
pvbus_m_range	PVBus	Master	Specifies the address range for the bus master. The range must be 4KB aligned and a multiple of 4KB in size. If the address range is larger than the size of the slave device, the slave is aliased.
pvbus_s	PVBus	Slave	Accepts incoming transactions. Connect this port to a bus master, or to the output of another bus decoder.

3.2.7 PVBusExclusiveMonitor

Global exclusive monitor. This model is written in C++.

PVBusExclusiveMonitor contains the following CADI targets:

- PVBusExclusiveMonitor

PVBusExclusiveMonitor contains the following MTI components:

- *4.102 PVBusExclusiveMonitor - trace* on page 4-2752
- *4.105 PVBusMapper - trace* on page 4-2758

Table 3-49 Ports

Name	Protocol	Type	Description
excl_cleared	*Signal* on page 2-85	Master	Exclusive monitor clear signal port.
pvbus_m	PVBus	Master	Bus master port.
pvbus_s	PVBus	Slave	Bus slave port.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-50 Parameters for PVBusExclusiveMonitor

Name	Type	Default value	Description
apply_access_width_criteria_to_non_excl_stores	bool	0x1	Apply the given exclusive store width matching criteria to non-exclusive stores
clear_on_strex_address_mismatch	bool	0x1	Whether monitor is cleared when strex fails due to address mismatch
enable_component	bool	0x1	Enable component
log2_granule_size	int	0x0	log2 of address granule size
match_access_width	bool	0x0	Fail STREX if not the same access width as LDREX
match_secure_state	bool	0x1	Treat the secure state like an address bit
monitor_access_level	int	0x0	0: Monitor all accesses, 1: Monitor all accesses except WriteBack, 2: Only monitor accesses with memory type NonCacheable or Device
monitor_non_excl_stores	bool	0x0	Monitor non-exclusive stores from the same master
number_of_monitors	int	0x8	Number of monitors
shareability_domain	int	0x3	Maximum shareability domain of interest, transactions outside of the domain will pass through un-monitored (0-non-shared, 1-inner, 2-outer, 3-system)

3.2.8 PVBusExclusiveSquasher

Squashes the exclusive attribute on bus transactions. This model is written in LISA+.

PVBusExclusiveSquasher contains the following CADI targets:

- PVBusExclusiveSquasher

PVBusExclusiveSquasher contains the following MTI components:

- *4.105 PVBusMapper - trace* on page 4-2758

Table 3-51 Ports

Name	Protocol	Type	Description
pvbus_m	PVBus	Master	Bus master port.
pvbus_s	PVBus	Slave	Bus slave port.

3.2.9 PVBusLogger

A PVBusLogger has a slave and a master port and traffic is passed straight through. All traffic is logged using an MTI trace event. This model is written in C++.

PVBusLogger contains the following CADI targets:

- PVBusLogger

PVBusLogger contains the following MTI components:

- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-52 Ports

Name	Protocol	Type	Description
pvbus_m	PVBus	Master	Bus master port.
pvbus_s	PVBus	Slave	Bus slave port.

Table 3-53 Parameters for PVBusLogger

Name	Type	Default value	Description
trace_debug	bool	0x0	Enable tracing of debug transactions
trace_snoops	bool	0x0	Enable tracing of ACE snoop requests

3.2.10 PVBusMapper

Allow transactions to be remapped arbitrarily. This model is written in C++.

PVBusMapper contains the following MTI components:

- *4.105 PVBusMapper - trace* on page 4-2758

About PVBusMapper

This component performs like PVBusModifier, but in addition:

- It has multiple downstream ports
- It allows routing of transactions to any one of these ports
- It allows arbitrary remapping of transaction addresses and attributes

As a generic modeling component, it does not have a hardware revision code.

For an example of how to use PVBusMapper, see $PVLIB_HOME/examples/LISAPlus/RemappingWithPVBusMapper/.

Table 3-54 Ports

Name	Protocol	Type	Description
control	PVBusMapperControl	Master	Configuration port to determine mappings.
pvbus_m[64]	PVBus	Master	Bus master ports.
pvbus_s	PVBus	Slave	Bus slave port.
reset	*Signal* on page 2-85	Slave	Reset signal.

3.2.11 PVBusMaster

The PVBusMaster subcomponent allows a device to generate PVBus transactions. This model is written in C++.

PVBusMaster contains the following MTI components:

- *4.106 PVBusMaster - trace* on page 4-2759

About PVBusMaster

If you want a component to act as a bus master, instantiate a PVBusMaster subcomponent and then use its control port to create one or more transaction generators to generate transactions on the bus. $PVLIB_HOME/examples/LISA/BusComponents/DmaTransfer.lisa is an example component that shows this.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-55 Ports

Name	Protocol	Type	Description
control	*PVTransactionMaster* on page 2-77	Slave	Enables the owning component to instantiate pv::TransactionGenerator objects.
pvbus_m	PVBus	Master	Sends out generated transactions to the bus.
reset	*Signal* on page 2-85	Slave	On the de-assert of this signal, a reset of the bus master will be latched this is used by the bus deadlock detection logic.

Related concepts

1.15.2 TransactionGenerator efficiency considerations on page 1-50

3.2.12 PVBusModifier

Allow transactions to be remapped arbitrarily. This model is written in C++.

PVBusModifier contains the following MTI components:

- *4.105 PVBusMapper - trace* on page 4-2758

Note

PVBusModifierx2 is an identical component, except it has two downstream ports.

Table 3-56 Ports

Name	Protocol	Type	Description
control	PVBusMapperControl	Master	Configuration port to determine mappings.
pvbus_m	PVBus	Master	Bus master port.
pvbus_s	PVBus	Slave	Bus slave port.
reset	*Signal* on page 2-85	Slave	Reset signal.

3.2.13 PVBusRouter

Allow transactions to be routed arbitrarily. This model is written in LISA+.

PVBusRouter contains the following CADI targets:

- PVBusRouter

PVBusRouter contains the following MTI components:

- *4.105 PVBusMapper - trace* on page 4-2758

Table 3-57 Ports

Name	Protocol	Type	Description
control	PVBusRouterControl	Master	Configuration port to determine filters.
pvbus_m[64]	PVBus	Master	Bus master ports.
pvbus_s	PVBus	Slave	Bus slave port.

3.2.14 PVBusSlave

A PVBusSlave handles incoming transactions, and handles support for mapping regions of device address space to work as RAM/ROM/device memory. This model is written in C++.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PVBusSlave contains the following MTI components:

- *4.107 PVBusSlave - trace* on page 4-2760

About PVBusSlave

Any component that acts as a bus slave must:

- Provide a PVBus slave port.
- Instantiate a PVBusSlave subcomponent, with the size parameter configured for the address range covered by the device.
- Connect the slave port to the `pvbus_s` port on the PVBusSlave.

By default, the PVBusSlave translates all transactions into `read()` and `write()` requests on the device port.

The control port enables you to configure the PVBusSlave for a device. This lets you define the behavior of the different regions of the address space on the device. You can configure regions separately for read and write, at a 4KB granularity. This permits you to set memory-like, device-like, abort, or ignore access address regions.

Transactions to memory-like regions are handled internally by the PVBusSlave subcomponent. Abort and ignore regions are also handled by the PVBusSlave.

Transactions to device-like regions are forwarded to the device port. Your device must implement the `read()` and `write()` methods on the device port if any regions are configured as device-like.

This component typically does not significantly affect the performance of a PV system. However, correct implementation of the PVBusSlave component is critical to the performance of the overall PV system. For example, routing a request to a PVDevice port is slower than letting the PVBusSlave component handle the request internally. Arm recommends using the internal support for memory-like regions where possible.

Table 3-58 Ports

Name	Protocol	Type	Description
`control`	*`PVBusSlaveControl` on page 2-75*	Slave	Enables the owning component to control which regions of the device memory are to be handled as RAM/ROM/Device. These settings can be changed dynamically. For example, when a Flash component is being programmed, it can switch to treating reads as Device requests instead of ROM requests.
`device`	*`PVDevice` on page 2-77*	Master	Passes on requests for peripheral register accesses to permit the owning component to handle the request.
`pvbus_s`	`PVBus`	Slave	Handles incoming requests from bus masters.
`reset`	*`Signal` on page 2-85*	Slave	On the assert of this signal, a reset of the bus slave will be latched this is used by the bus deadlock detection logic.

3.2.15 PVWriteBuffer

The PVWriteBuffer subcomponent buffers PVBus transactions. This model is written in C++.

PVWriteBuffer contains the following MTI components:

- *4.105 PVBusMapper - trace* on page 4-2758
- *4.109 PVWriteBuffer - trace* on page 4-2767

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-59 Ports

Name	Protocol	Type	Description
barrier_notify_s	PVWriteBuffer_BarrierPort	Slave	Barrier notification input.
clk_in	ClockSignal	Slave	Clock input.
pvbus_m	PVBus	Master	Master connection to memory bus.
pvbus_s	PVBus	Slave	Slave connection for transactions to be buffered.
reset_in	*Signal* on page 2-85	Slave	Reset input.
serror_notify_m	PVWriteBuffer_SErrorPort	Master	SError output generation.

3.2.16 SimplePVBusMaster

Component to generate PVTransactions with configurable attributes and address. This model is written in LISA+.

SimplePVBusMaster contains the following CADI targets:

- ClockTimerThread64
- SchedulerThread
- SchedulerThreadEvent
- SimplePVBusMaster

SimplePVBusMaster contains the following MTI components:

- *4.106 PVBusMaster - trace* on page 4-2759
- *4.107 PVBusSlave - trace* on page 4-2760

Table 3-60 Ports

Name	Protocol	Type	Description
pvbus_m	PVBus	Master	Output of generated transactions.
pvbus_s	PVBus	Slave	-

Table 3-61 Parameters for SimplePVBusMaster

Name	Type	Default value	Description
verbose	bool	0x0	verbose

3.2.17 TZSwitch

Allow TrustZone secure/normal bus signals to be routed separately. This model is written in LISA+.

TZSwitch contains the following CADI targets:

- TZSwitch

TZSwitch contains the following MTI components:

- *4.105 PVBusMapper - trace* on page 4-2758

About TZSwitch

The default behavior is to forward secure transactions to pvbus_port_a, and normal transactions to pvbus_port_b.

You must only use TZSwitches if the routing decisions change infrequently, for example as part of a memory remap.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

The `secure` and `normal` parameters control the initial state of this component:

0 Ignore these transactions.

1 Forward the transactions to `pvbus_port_a`.

2 Forward the transactions to `pvbus_port_b`.

3 Generate an abort for these transactions.

The numbers used for initial configuration are not the same as the enumeration constants used to control routing at runtime.

Table 3-62 Ports

Name	Protocol	Type	Description
`control`	`TZSwitchControl`	Slave	Controls routing of transactions.
`pvbus_input`	`PVBus`	Slave	Slave port for connection to PVBus master/decoder.
`pvbus_port_a`	`PVBus`	Master	Output port a.
`pvbus_port_b`	`PVBus`	Master	Output port b.

Table 3-63 Parameters for TZSwitch

Name	Type	Default value	Description
`normal`	`int`	`0x2`	Normal Port
`secure`	`int`	`0x1`	Secure Port

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

3.3 Clocking components

This section describes the Clocking components.

The clocking components and protocols provide a mechanism for systems to regulate the execution rate of components. Clocking includes the concept of clock rates, dividers to change clock rates, and timers to generate callbacks based on those clock rates.

If the MasterClock component is instantiated in a system, it provides a consistent master clock rate. Although this rate is not defined, you can consider this to be 1Hz, even for non-SystemC systems. ClockDivider components are able to convert this clock rate into a new rate using a multiplier and divider, although the clock rate cannot be divided to be less than 1Hz. You can cascade ClockDivider components to produce many different clock rates within a system. The maximum ratio of any two clocks in the system must be less than 2^{32}.

ClockTimer components can be instantiated by a component and connected to any MasterClock or ClockDivider output. ClockTimers can generate callbacks after a given number of ticks of that clock. ClockTimers can invoke a behavior on the component to permit the component to perform work. The component can then request the ClockTimer to repeat its count.

This section contains the following subsections:

- *3.3.1 ClockDivider* on page 3-113.
- *3.3.2 ClockTimer* on page 3-114.
- *3.3.3 ClockTimer64* on page 3-114.
- *3.3.4 ClockTimerThread* on page 3-115.
- *3.3.5 ClockTimerThread64* on page 3-115.
- *3.3.6 MasterClock* on page 3-116.

3.3.1 ClockDivider

A ClockDivider is a library component that takes a ClockSignal on its input port (which could come from the output of a MasterClock, or from another ClockDivider), and generates a new ClockSignal on its output port, representing a clock frequency that is related to the input clock by the ratio of the multiply and divide parameters. This model is written in C++.

ClockDivider contains the following CADI targets:

- ClockDivider

ClockDivider contains the following MTI components:

- *4.60 ClockDivider - trace* on page 4-2438

About ClockDivider

This component uses a configurable ratio to convert the ClockSignal rate at its input to a new ClockSignal rate at its output. Changes to the input rate or ratio take effect immediately and clocking components dependent on the output rate continue counting at the new rate.

This component does not normally incur a runtime performance cost. However, reprogramming the clock rate causes all related clocks and timers to be recalculated.

For an example of the use of ClockDividers, see the `VEMotherBoard.lisa` component in the `$PVLIB_HOME/examples/LISA/FVP_VE/LISA/` directory of your Fast Models distribution.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-64 Ports

Name	Protocol	Type	Description
clk_in	ClockSignal	Slave	Input clock signal, coming from a MasterClock or another ClockDivider.
clk_out	ClockSignal	Master	Clock signal generated by this ClockDivider.
rate	*ClockRateControl* on page 2-65	Slave	Permits you to dynamically change the clock divider ratio.

3.3.2 ClockTimer

A ClockTimer provides support for counting a number of ticks at the rate determined by the input clock. When the countdown reaches zero, the timer can send a signal to another component. That component can return a value that will cause the timer to start counting down again. Setting up a timer is very efficient, and no host processing time is used while a counter is counting down: when a timer is started, the scheduler precomputes the finish time. To use a ClockTimer connect the ClockTimer's 'timer_callback' port to a slave port that implements the 'signal()' behaviour and connect a clock signal to the clk_in port. Use the 'timer_control' port to start the timer counting down for a given number of ticks. This model is written in C++.

About ClockTimer

This component provides a mechanism for other components to schedule a callback after a number of ticks at a given ClockSignal rate.

An active ClockTimer component incurs no simulation overhead. For best performance, avoid having your performance-critical code frequently cancel timers or query the number of remaining ticks.

Table 3-65 Ports

Name	Protocol	Type	Description
clk_in	ClockSignal	Slave	Determines the tick rate of the timer.
timer_callback	*TimerCallback* on page 2-65	Master	Port on which a signal is sent after the number of scheduled ticks has elapsed.
timer_control	*TimerControl* on page 2-65	Slave	Permits the timer to be set, canceled and queried.

3.3.3 ClockTimer64

A ClockTimer64 provides support for counting a number of ticks at the rate determined by the input clock. When the countdown reaches zero, the timer can send a signal to another component. That component can return a value that will cause the timer to start counting down again. Setting up a timer is very efficient, and no host processing time is used while a counter is counting down: when a timer is started, the scheduler precomputes the finish time. This version of the timer provides 64 bit resolution. This model is written in C++.

About ClockTimer64

This component provides a mechanism for other components to schedule a callback after a number of ticks at a given ClockSignal rate.

An active ClockTimer64 component incurs no simulation overhead. For best performance, avoid having your performance-critical code frequently cancel timers or query the number of remaining ticks.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-66 Ports

Name	Protocol	Type	Description
clk_in	ClockSignal	Slave	Determines the tick rate of the timer.
timer_callback	*TimerCallback64 on page 2-65*	Master	Port on which a signal is sent after the number of scheduled ticks has elapsed.
timer_control	*TimerControl64 on page 2-66*	Slave	Permits the timer to be set, canceled and queried.

3.3.4 ClockTimerThread

A ClockTimerThread(64) is a drop-in replacement for a ClockTimer(64) component. The main difference to the ClockTimer component is that the ClockTimerThread runs the signal() callback from a proper scheduler thread. This mean that the signal() function may directly or indirectly invoke wait() functions to wait for time or events. This is not allowed for the ClockTimer component which does not use a thread. Components which issue bus transactions from within the timer signal() callback must use ClockTimerThread(64) rather than ClockTimer(64). This model is written in LISA+.

ClockTimerThread contains the following CADI targets:

- ClockTimerThread
- ClockTimerThread64
- SchedulerThread
- SchedulerThreadEvent

Table 3-67 Ports

Name	Protocol	Type	Description
clk_in	ClockSignal	Slave	Determines the tick rate of the timer.
timer_callback	*TimerCallback on page 2-65*	Master	Port on which a signal is sent after the number of scheduled ticks has elapsed.
timer_control	*TimerControl on page 2-65*	Slave	Permits the timer to be set, canceled and queried.

3.3.5 ClockTimerThread64

A ClockTimerThread64 is a drop-in replacement for ClockTimer64. The main difference to the ClockTimer component is that the ClockTimerThread runs the signal() callback from a proper scheduler thread. This mean that the signal() function may directly or indirectly invoke wait() functions to wait for time or events. This is not allowed for the ClockTimer component which does not use a thread. Components which issue bus transactions from within the timer signal() callback must use ClockTimerThread(64) rather than ClockTimer(64). This model is written in LISA+.

ClockTimerThread64 contains the following CADI targets:

- ClockTimerThread64
- SchedulerThread
- SchedulerThreadEvent

Table 3-68 Ports

Name	Protocol	Type	Description
clk_in	ClockSignal	Slave	Determines the tick rate of the timer.
timer_callback	*TimerCallback64 on page 2-65*	Master	Port on which a signal is sent after the number of scheduled ticks has elapsed.
timer_control	*TimerControl64 on page 2-66*	Slave	Permits the timer to be set, canceled and queried.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

3.3.6 MasterClock

A source of a clock signal representing the base clock rate of the simulation (nominally 1Hz). The signal from the output port can be connected to a ClockDivider, to generate clock signals at a different clock rate. The output can also be connected to a ClockTimer in order to generate scheduled events, or to any other components that accept a ClockSignal input. See ClockSignal.lisa for more information. This model is written in C++.

About MasterClock

This component provides a single ClockSignal output that can be used to drive the ClockSignal input of ClockDividers, ClockTimers and other clocking components.

The rate of the MasterClock is not defined because all clocking is relative, but can be considered to be 1Hz.

A system might contain more than one MasterClock, all of which generate the same ClockSignal rate.

Table 3-69 Ports

Name	Protocol	Type	Description
clk_out	ClockSignal	Master	Master clock rate.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

3.4 Core components

This section describes the Core components.

Code Translation (CT) components translate instructions on the fly and cache the translations to enable fast execution of code, but sacrifice timing accuracy. They also use efficient PV bus models to enable fast access to memory and devices.

CT components implement most of the processor features but differ in some ways:

- They do not model cycle timing. In aggregate, all instructions execute in one processor master clock cycle, except for Wait For Interrupt.
- Write buffers are not modeled on all processors.
- Most aspects of TLB behavior are implemented in the models. In Armv7 models and later, the TLB memory attribute settings are used when stateful cache is enabled.
- No device-accurate MicroTLB is implemented.
- Device-accurate modeling of multiple TLBs is off by default.
- A single memory access port is implemented. The port combines accesses for instruction, data, DMA, and peripherals. Configuration of the peripheral port memory map register is ignored.
- All memory accesses are atomic and are performed in *Programmer's View* (PV) order. Unaligned accesses are always performed as byte transfers.
- Some instruction sequences are executed atomically so that system time does not advance during their execution. This difference in behavior can affect sequential accesses of device registers where devices are expecting time to move on between each access.
- Interrupts are not taken at every instruction boundary.
- Integration and test registers are not implemented.
- Models do not support running Software Test Libraries (STLs).
- Not all CP14 debug registers are implemented on all processors.
- Breakpoint types that the models support directly are:
  - Single address unconditional instruction breakpoints.
  - Single address unconditional data breakpoints.
  - Unconditional instruction address range breakpoints.
- Pseudoregisters in the debugger support processor exception breakpoints. Setting an exception register to a nonzero value stops execution on entry to the associated exception vector.
- Cluster models do not simulate all cores at the same time. They execute a number of instructions on each core in turn. There can be a bias in the order in which cores run after a restart (for example, core 0 always runs first), so the simulation might hit breakpoints on the favored core more often.
- Performance counters are not implemented on all models.
- Some models implement caches, although all processor models implement cache control registers.
- ECC and parity schemes are hardware-specific so are not modeled.
- Models use a simplified view of the external buses.

This section contains the following subsections:

- *3.4.1 AEMvACT* on page 3-118.
- *3.4.2 ARMAEMv8AMPCT* on page 3-154.
- *3.4.3 ARMAEMv8A_AEMv8AMPCT* on page 3-189.
- *3.4.4 ARMAEMv8MCT* on page 3-252.
- *3.4.5 ARMCortexA5CT* on page 3-273.
- *3.4.6 ARMCortexA5MPx1CT* on page 3-277.
- *3.4.7 ARMCortexA7x1CT* on page 3-281.
- *3.4.8 ARMCortexA8CT* on page 3-288.
- *3.4.9 ARMCortexA9MPx1CT* on page 3-292.
- *3.4.10 ARMCortexA9UPCT* on page 3-298.
- *3.4.11 ARMCortexA15x1CT* on page 3-302.
- *3.4.12 ARMCortexA17x1CT* on page 3-308.
- *3.4.13 ARMCortexA32x1CT* on page 3-314.
- *3.4.14 ARMCortexA35x1CT* on page 3-323.
- *3.4.15 ARMCortexA53x1CT* on page 3-332.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

- *3.4.16 ARMCortexA55CT* on page 3-342.
- *3.4.17 ARMCortexA55CT_CortexA75CT* on page 3-352.
- *3.4.18 ARMCortexA55CT_CortexA76CT* on page 3-363.
- *3.4.19 ARMCortexA57x1CT* on page 3-374.
- *3.4.20 ARMCortexA65AECT* on page 3-384.
- *3.4.21 ARMCortexA65AECT_CortexA76AECT* on page 3-394.
- *3.4.22 ARMCortexA65CT* on page 3-406.
- *3.4.23 ARMCortexA72x1CT* on page 3-416.
- *3.4.24 ARMCortexA73x1CT* on page 3-425.
- *3.4.25 ARMCortexA75CT* on page 3-434.
- *3.4.26 ARMCortexA76AECT* on page 3-444.
- *3.4.27 ARMCortexA76CT* on page 3-454.
- *3.4.28 ARMCortexA77CT* on page 3-464.
- *3.4.29 ARMCortexA78CCT* on page 3-475.
- *3.4.30 ARMCortexA78CT* on page 3-485.
- *3.4.31 ARMCortexM0CT* on page 3-495.
- *3.4.32 ARMCortexM0PlusCT* on page 3-497.
- *3.4.33 ARMCortexM3CT* on page 3-499.
- *3.4.34 ARMCortexM4CT* on page 3-502.
- *3.4.35 ARMCortexM7CT* on page 3-505.
- *3.4.36 ARMCortexM23CT* on page 3-509.
- *3.4.37 ARMCortexM33CT* on page 3-513.
- *3.4.38 ARMCortexM35PCT* on page 3-519.
- *3.4.39 ARMCortexM55CT* on page 3-525.
- *3.4.40 ARMCortexR4CT* on page 3-535.
- *3.4.41 ARMCortexR5x1CT* on page 3-538.
- *3.4.42 ARMCortexR7x1CT* on page 3-541.
- *3.4.43 ARMCortexR8x1CT* on page 3-545.
- *3.4.44 ARMCortexR52x1CT* on page 3-550.
- *3.4.45 ARMCortexX1CT* on page 3-564.
- *3.4.46 ARMNeoverseE1CT* on page 3-575.
- *3.4.47 ARMNeoverseN1CT* on page 3-585.
- *3.4.48 ARMSC000CT* on page 3-595.
- *3.4.49 ARMSC300CT* on page 3-597.

3.4.1 AEMvACT

ARM AEM A-Profile(MP) CPU component - number of cores configurable at runtime. This model is written in C++.

AEMvACT contains the following CADI targets:

- ARM_AEM-A_MP
- Cluster_ARM_AEM-A_MP
- PVCache
- TlbCadi

AEMvACT contains the following MTI components:

- *4.3 ARM_AEM-A_MP - trace* on page 4-870
- *4.48 ARMv8Cluster - trace* on page 4-2401
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402
- *4.68 DSU - trace* on page 4-2463
- *4.73 GICv3CPUInterfaceDecoder - trace* on page 4-2489
- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.108 PVCache - trace* on page 4-2761
- *4.114 TLB - trace* on page 4-2834

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

About AEMvACT

This component is an AEM that implements all architectural features in Armv8-A and in addition implements the following Future Architecture Technologies:

- Scalable Vector Extension version 2 (SVE2)
- Transactional Memory Extension (TME)
- Call Stack Recorder Extension (CSRE)
- Branch Record Buffer Extension (BRBE)

Mapping architectural features to Fast Models releases

The following table shows which Fast Models releases the key architectural features are implemented in, from Armv8.3-A onwards. Support is at EAC quality unless otherwise stated.

Table 3-70 Architectural features and Fast Models releases

Architecture version	Feature	Fast Models version									
		11.13	11.12	11.11	11.10	11.9	11.8	11.7	11.6	11.5	11.4
Armv8.3-A	All [b]	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Armv8.4-A	All [c]	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Armv8.5-A	Memory Tagging Extension (MTE) functionality	✓	✓	✓	✓	✓	✓	✓	✓ (Beta)	-	-
	MTE performance	✓	✓	✓	✓	✓	✓ (Beta)	-	-	-	-
	All except MTE [d].	✓	✓	✓	✓	✓	✓	✓	✓	✓	-
Armv8.6-A	All [e]	✓	✓	✓	✓	✓	-	-	-	-	-
Armv8.7-A	Limited TLBI maintenance	✓ (Alpha)	-	-	-	-	-	-	-	-	-
	All except TLBI maintenance [f]	✓	-	-	-	-	-	-	-	-	-
Future Architecture Technologies	SVE2, TME, CSRE, BRBE. [g]	✓	✓	✓	✓	✓	✓	-	-	-	-

[b] Armv8.3-A includes pointer authentication, nested virtualization, Advanced SIMD complex number support, improved Javascript data type conversion support, memory consistency model changes, support for larger system-visible caches, debug over powerdown.

[c] Armv8.4-A includes Secure EL2, additional cryptographic hashes, activity monitors, enhanced support for nested virtualization, Memory Partitioning and Monitoring (MPAM), stage 2 forced write-back, dot product instructions, data independent timing instructions, Large System Extensions (LSE), support for TLB maintenance instructions and for TLB range instructions, Translation Table Level (TTL), enhancements to weaker release consistency, debug relaxations and extensions, flag manipulation instructions.

[d] Armv8.5-A includes Branch Target Indicators (BTI), Enhanced Virtualization Traps (EVT), Random Number Generator (RNG), Flag Manipulation instructions, Floating-point round to integer, DC CVADP.

[e] Armv8.6-A includes General Matrix Multiply (GEMM), EnhancedPAC2, FPAC, Data Gathering Hint, Additional PMU events, Enhanced Counter Virtualization.

[f] Armv8.7-A includes WFE and WFI with timeouts, Larger physical address for 4KB and 16KB translation granules, MTE Asymmetric Fault Handling, Enhancements to Privilege Access Never (PAN), Enhancements to PMU and SPE.

[g] Implemented by AEMvA only.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-71 Ports

Name	Protocol	Type	Description
CNTHPIRQ[8]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTHPSIRQ[8]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTHVIRQ[8]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTHVSIRQ[8]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTPNSIRQ[8]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTPSIRQ[8]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTVIRQ[8]	*Signal* on page 2-85	Master	Timer signals to SOC.
CRITICALIRQ[8]	*Signal* on page 2-85	Master	RAS Critical Error Interrupt.
ERRORIRQ[8]	*Signal* on page 2-85	Master	RAS Error Recovery Interrupt.
FAULTIRQ[8]	*Signal* on page 2-85	Master	RAS Fault Handling Interrupt
acp_s	PVBus	Slave	AXI ACP slave port.
broadcastcachemaint	*Signal* on page 2-85	Slave	ACE defined pins.
broadcastinner	*Signal* on page 2-85	Slave	ACE defined pins.
broadcastouter	*Signal* on page 2-85	Slave	ACE defined pins.
cfgend[8]	*Signal* on page 2-85	Slave	This signal if for EE bit initialisation.
cfgsdisable	*Signal* on page 2-85	Slave	This signal disables write access to some secure Interrupt Controller registers.
clk_in	ClockSignal	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the core executes instructions.
clrexmonack	*Signal* on page 2-85	Master	Acknowledge handshake signal for the clrexmonreq signal
clrexmonreq	*Signal* on page 2-85	Slave	Signals the clearing of an external global exclusive monitor
clusterid	*Value* on page 2-85	Slave	The port reads the value in CPU ID register field, bits[11:8] of the MPIDR.
cntvalueb	*CounterInterface* on page 2-84	Slave	Interface to SoC level counter module.
commirq[8]	*Signal* on page 2-85	Master	Interrupt signal from debug communications channel.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-71 Ports (continued)

Name	Protocol	Type	Description
config64[8]	*Signal* on page 2-85	Slave	Register width after reset.
cp15sdisable[8]	*Signal* on page 2-85	Slave	This signal disables write access to some system control processor registers.
cpuporeset[8]	*Signal* on page 2-85	Slave	CPU power on reset. Initializes all the processor logic, including debug logic.
cryptodisable[8]	*Signal* on page 2-85	Slave	Disable cryptography extensions after reset.
cti[8]	*v8EmbeddedCrossTrigger_controlprotocol* on page 2-84	Master	Cross trigger matrix port.
cti0extin[4]	*Signal* on page 2-85	Slave	CTI trace inputs for core 0.
cti0extout[4]	*Signal* on page 2-85	Master	CTI trace outputs for core 0.
cti1extin[4]	*Signal* on page 2-85	Slave	CTI trace inputs for core 1.
cti1extout[4]	*Signal* on page 2-85	Master	CTI trace outputs for core 1.
cti2extin[4]	*Signal* on page 2-85	Slave	CTI trace inputs for core 2.
cti2extout[4]	*Signal* on page 2-85	Master	CTI trace outputs for core 2.
cti3extin[4]	*Signal* on page 2-85	Slave	CTI trace inputs for core 3.
cti3extout[4]	*Signal* on page 2-85	Master	CTI trace outputs for core 3.
cti4extin[4]	*Signal* on page 2-85	Slave	CTI trace inputs for core 4.
cti4extout[4]	*Signal* on page 2-85	Master	CTI trace outputs for core 4.
cti5extin[4]	*Signal* on page 2-85	Slave	CTI trace inputs for core 5.
cti5extout[4]	*Signal* on page 2-85	Master	CTI trace outputs for core 5.
cti6extin[4]	*Signal* on page 2-85	Slave	CTI trace inputs for core 6.
cti6extout[4]	*Signal* on page 2-85	Master	CTI trace outputs for core 6.
cti7extin[4]	*Signal* on page 2-85	Slave	CTI trace inputs for core 7.
cti7extout[4]	*Signal* on page 2-85	Master	CTI trace outputs for core 7.
ctidbgirq[8]	*Signal* on page 2-85	Master	Cross Trigger Interface (CTI) interrupt trigger output.
dbgen[8]	*Signal* on page 2-85	Slave	External debug interface.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-71 Ports (continued)

Name	Protocol	Type	Description
dbgnopwrdwn[8]	*Signal* on page 2-85	Master	Debug no power down request.
dbgpwrdwnack[8]	*Signal* on page 2-85	Master	Debug power down acknowledge.
dbgpwrdwnreq[8]	*Signal* on page 2-85	Slave	Debug power down request.
dbgpwrupreq[8]	*Signal* on page 2-85	Master	Debug power up request.
dev_debug_s	PVBus	Slave	External debug interface.
etm[8]	PVBus	Master	Embedded trace macrocell port.
event	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE.
external_trace_reset[8]	*Signal* on page 2-85	Slave	ETMv4 External Trace Reset signal.
fiq[8]	*Signal* on page 2-85	Slave	This signal drives the CPUs fast-interrupt handling.
gicv3_redistributor_s[8]	*GICv3Comms* on page 2-84	Slave	GICv3 cpu interface ports.
irq[8]	*Signal* on page 2-85	Slave	This signal drives the CPUs interrupt handling.
irqs[224]	*Signal* on page 2-85	Slave	These signals drive the CPU's interrupt controller interrupt lines.
l2reset	*Signal* on page 2-85	Slave	This signal resets timer and interrupt controller.
memorymapped_debug_s	PVBus	Slave	External debug interface.
niden[8]	*Signal* on page 2-85	Slave	External debug interface.
periphbase	*Value_64* on page 2-85	Slave	This port sets the base address of private peripheral region.
pmbirq[8]	*Signal* on page 2-85	Master	Interrupt signal from the statistical profiling unit.
pmuirq[8]	*Signal* on page 2-85	Master	Interrupt signal from performance monitoring unit.
presetdbg[8]	*Signal* on page 2-85	Slave	Initialize the shared debug APB, Cross Trigger Interface (CTI), and Cross Trigger Matrix (CTM) logic.
pvbus_m0	PVBus	Master	The core will generate bus requests on this port.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-71 Ports (continued)

Name	Protocol	Type	Description
rei[8]	*Signal* on page 2-85	Slave	Individual processor RAM Error Interrupt signal input.
reset[8]	*Signal* on page 2-85	Slave	Raising this signal will put the core into reset mode.
romaddr	*Value_64* on page 2-85	Slave	Debug ROM base address.
romaddrv	*Signal* on page 2-85	Slave	Debug ROM base address valid.
rvbar[8]	*Value_64* on page 2-85	Slave	Reset vector base address.
sei[8]	*Signal* on page 2-85	Slave	Per core System Error physical pins.
smpnamp[8]	*Signal* on page 2-85	Master	This signals AMP or SMP mode for each core.
spiden[8]	*Signal* on page 2-85	Slave	External debug interface.
spniden[8]	*Signal* on page 2-85	Slave	External debug interface.
standbywfe[8]	*Signal* on page 2-85	Master	This signal indicates if a core is in WFE state.
standbywfi[8]	*Signal* on page 2-85	Master	This signal indicates if a core is in WFI state.
standbywfil2	*Signal* on page 2-85	Master	This signal indicated all cores and L2 are in a power down state
teinit[8]	*Signal* on page 2-85	Slave	This signal provides default exception handling state.
ticks[8]	*InstructionCount* on page 2-84	Master	This port should be connected to one of the two ticks ports on a 'visualisation' component, in order to display a running instruction count.
trace_unit_reset[8]	*Signal* on page 2-85	Slave	ETMv4 Trace Unit Reset signal.
trbirq[8]	*Signal* on page 2-85	Master	-
vcpumntirq[8]	*Signal* on page 2-85	Master	Interrupt signal for virtual CPU maintenance IRQ.
vfiq[8]	*Signal* on page 2-85	Slave	Virtual FIQ input. Note that the irq/fiq pins are wired directly to the core if there is no internal VGIC. If there is an internal VGIC then these are ignored.
vinithi[8]	*Signal* on page 2-85	Slave	This signal controls of the location of the exception vectors at reset.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-71 Ports (continued)

Name	Protocol	Type	Description
virq[8]	*Signal* on page 2-85	Slave	Virtual IRQ input. Note that the irq/fiq pins are wired directly to the core if there is no internal VGIC. If there is an internal VGIC then these are ignored.
virtio_s	PVBus	Slave	The virtio coherent port, hooks directly into the L2 system and becomes coherent (assuming attributes are set correctly).
vsei[8]	*Signal* on page 2-85	Slave	Processor Virtual System Error Interrupt request.

Table 3-72 Parameters for ARM_AEM-A_MP

Name	Type	Default value	Description
cpu0.CFGEND	bool	0x0	Endianness configuration at reset. 0, little endian. 1, big endian.
cpu0.CONFIG64	bool	0x1	Register width configuration at reset. 0, AArch32. 1, AArch64.
cpu0.CP15SDISABLE	bool	0x0	Initialize to disable access to some CP15 registers
cpu0.CP15SDISABLE2	bool	0x0	Initialize to disable access to some CP15 registers
cpu0.CRYPTODISABLE	bool	0x0	Disable cryptographic features.
cpu0.DCZID-log2-block-size	int	0x8	Log2 of the block size cleared by DC ZVA instruction (as read from DCZID_EL0).
cpu0.DCZVA_single_write	bool	0x0	Execute the DCZVA as a single write
cpu0.MPIDR-override	int	0x0	Override of MPIDR value. If nonzero will override the MT, cluster and CPU ID bits in MPIDR.
cpu0.RVBAR	int	0x0	Value of RVBAR_ELx register.
cpu0.RVBAR32	int	0x0	Reset vector address in AARCH32 when VINITHI is not set and ignore_rvbar_in_aarch32 is set
cpu0.SMPnAMP	bool	0x1	Enable broadcast messages necessary for correct SMP operation at reset.
cpu0.TEINIT	bool	0x0	Instruction set state when resetting into AArch32. 0, A32. 1, T32.
cpu0.VINITHI	bool	0x0	Reset value of SCTLR.V.
cpu0.ase-present	bool	0x1	Set whether the model has been built with NEON support
cpu0.clock_divider	int	0x1	Clock divider ratio for asymmetric MP clocking.
cpu0.clock_multiplier	int	0x1	Clock divider ratio for asymmetric MP clocking.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-72 Parameters for ARM_AEM-A_MP (continued)

Name	Type	Default value	Description
cpu0.crypto_aes	int	0x2	AES instructions supported (requires CryptoPlugin to be loaded). 0, not implemented. 1, AES instructions implemented. 2, AES and PMULL instructions implemented.
cpu0.crypto_sha1	int	0x1	SHA-1 instructions supported (requires CryptoPlugin to be loaded). 0, not implemented. 1, SHA1 instructions implemented.
cpu0.crypto_sha256	int	0x1	SHA-256 instructions supported (requires CryptoPlugin to be loaded). 0, not implemented. 1, SHA256 instructions implemented.
cpu0.crypto_sha3	int	0x0	Implement ARMv8.4 SHA-3 instructions (requires CryptoPlugin to be loaded). Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
cpu0.crypto_sha512	int	0x0	Implement ARMv8.4 SHA-512 instructions (requires CryptoPlugin to be loaded). Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
cpu0.crypto_sm3	int	0x0	Implement ARMv8.4 SM-3 instructions (requires CryptoPlugin to be loaded). Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
cpu0.crypto_sm4	int	0x0	Implement ARMv8.4 SM-4 instructions (requires CryptoPlugin to be loaded). Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
cpu0.cti-intack_mask	int	0x1	Set bits represent that the corresponding trigger requires software acknowledge via CTIINTACK
cpu0.cti-number_of_claim_bits	int	0x0	Number of implemented bits in CTICLAIMSET
cpu0.cti-number_of_triggers	int	0x8	Number of cti event triggers (default: 8, valid values: {3, 8-32})
cpu0.enable_crc32	int	0x0	CRC32 instructions supported. 0, not implemented. 1, CRC32 instructions implemented.
cpu0.enable_trace_special_hlt_imm16	bool	0x0	Enable usage of parameter trace_special_hlt_imm16
cpu0.etm-present	bool	0x1	Set whether the model has ETM support
cpu0.force-fpsid	bool	0x0	Override the FPSID value
cpu0.force-fpsid-value	int	0x0	Value to override the FPSID value to
cpu0.has_hcptr_tase	bool	0x1	If false, HCPTR.TASE is RES0

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-72 Parameters for ARM_AEM-A_MP (continued)

Name	Type	Default value	Description
cpu0.max_code_cache_mb	int	0x100	Maximum size of the simulation code cache (MiB). For platforms with more than 2 cores this limit will be scaled down. (e.g 1/8 for 16 or more cores)
cpu0.min_sync_level	int	0x0	Force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)
cpu0.number-of-breakpoints	int	0x10	Number of breakpoints.
cpu0.number-of-context-breakpoints	int	0x10	Number of breakpoints that are context aware.
cpu0.number-of-watchpoints	int	0x10	Number of watchpoints.
cpu0.operation_bandwidth	int	0x1	Operation width for ARMv8.4 PMU extension
cpu0.semihosting-A32_HLT	int	0xf000	A32 HLT number for semihosting calls.
cpu0.semihosting-A64_HLT	int	0xf000	A64 HLT number for semihosting calls.
cpu0.semihosting-ARM_SVC	int	0x123456	A32 SVC number for semihosting calls.
cpu0.semihosting-T32_HLT	int	0x3c	T32 HLT number for semihosting calls.
cpu0.semihosting-Thumb_SVC	int	0xab	T32 SVC number for semihosting calls.
cpu0.semihosting-cmd_line	string	""	Command line available to semihosting calls.
cpu0.semihosting-cwd	string	""	Base directory for semihosting file access.
cpu0.semihosting-enable	bool	0x1	Enable semihosting SVC/HLT traps.
cpu0.semihosting-heap_base	int	0x0	Virtual address of heap base.
cpu0.semihosting-heap_limit	int	0xf000000	Virtual address of top of heap.
cpu0.semihosting-prefix	bool	0x0	Prefix semihosting output with target instance name
cpu0.semihosting-stack_base	int	0x10000000	Virtual address of base of descending stack.
cpu0.semihosting-stack_limit	int	0xf000000	Virtual address of stack limit.
cpu0.semihosting-stderr_istty	bool	0x1	Result for semihost istty call when argument is stderr
cpu0.semihosting-stdin_istty	bool	0x1	Result for semihost istty call when argument is stdin
cpu0.semihosting-stdout_istty	bool	0x1	Result for semihost istty call when argument is stdout
cpu0.semihosting-use_stderr	bool	0x0	Send stderr from the simulated process to host stderr
cpu0.trace_special_hlt_imm16	int	0xf000	For this HLT number, IF enable_trace_special_hlt_imm16=true, skip performing usual HLT execution but call MTI trace if registered
cpu0.unpredictable_WPMASKANDBAS	int	0x1	Constrained unpredictable handling of watchpoints when mask and BAS fields specified. 0, IGNOREMASK. 1, IGNOREBAS (default). 2, REPEATBAS8. 3, REPEATBAS.
cpu0.vfp-enable_at_reset	bool	0x0	Enable VFP in CPACR, CPPWR, NSACR at reset. Warning: Arm recommends going through the implementation's suggested VFP power-up sequence!
cpu0.vfp-present	bool	0x1	Set whether the model has VFP support
cpu0.vfp-traps	bool	0x1	Implement support for trapping floating-point exceptions

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-72 Parameters for ARM_AEM-A_MP (continued)

Name	Type	Default value	Description
cpu0.vfp-traps-show-all	bool	0x0	Report all trapped floating-point exceptions in the syndrome when a combination occurs.
cpu0.wfet_early_or_delayed_timeout	int	0x0	WFET early or delayed timeout beyond the threshold value of CNTVCT_EL0 in percentage.
cpu0.wfit_early_or_delayed_timeout	int	0x0	WFIT early or delayed timeout beyond the threshold value of CNTVCT_EL0 in percentage.

Table 3-73 Parameters for Cluster_ARM_AEM-A_MP

Name	Type	Default value	Description
AIDR	int	0x0	Value of AIDR_EL1 register.
AMPIDR	int	0x400bbb000	Value of AMU Peripheral Identification Register
BPIMVA_causes_translation_lookup	bool	0x0	Do a translation when BPIMVA instruction is executed (which may cause a translation fault).
CCSIDR-L1D_override	int	0x0	If nonzero, override the value presented in CCSIDR for L1D (this is cosmetic and does not affect cache behaviour).
CCSIDR-L1I_override	int	0x0	If nonzero, override the value presented in CCSIDR for L1I (this is cosmetic and does not affect cache behaviour).
CCSIDR-L2_override	int	0x0	If nonzero, override the value presented in CCSIDR for L2 (this is cosmetic and does not affect cache behaviour).
CCSIDR-L3_override	int	0x0	If nonzero, allow L3 selection in CSSELR and present this value in CCSIDR (this is cosmetic and does not affect cache behaviour).
CPUCFR	int	0x0	Value of CPU Configuration Register
CTIPIDR	int	0x0	If non-zero, override the CTI Peripheral Identification Register
DBGPIDR	int	0x0	If non-zero, override the Debug Peripheral Identification Register
ERXMISC0_mask	int	0x0	Write Mask for ERXMISC0 RAS Register
GMID-log2-block-size	int	0x4	Log2 of the block size accessed by STGM/LDGM/STZGM instructions
ISV_set_to_0_for_stage2_synch_external_abort	bool	0x0	Whether ESR_EL2.ISV is set to 0 on stage 2 synchronous external aborts
MIDR	int	0x410fd0f0	Value of MIDR_EL1 register.
NUM_CORES	int	0x1	Number of cores per cluster
PA_SIZE	int	0x28	Physical address range supported. For ARMv8.0 and ARMv8.1 this is limited to 48 bits.
PMUPIDR	int	0x0	If non-zero, override the PMU Peripheral Identification Register
abort_execution_from_device_memory	bool	0x0	Execution from device memory generates a prefetch abort.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-73 Parameters for Cluster_ARM_AEM-A_MP (continued)

Name	Type	Default value	Description
advsimd_bf16_support_level	int	0x0	Implement BFloat16 operations from ARMv8.6. AArch64 Advanced SIMD and FP BFloat16 instructions are automatically enabled when has_arm_v8-6 is true. - 0, Not implemented. - 1, AArch64 Advanced SIMD and FP BFloat16 instructions only. - 2, AArch32 Advanced SIMD and VFP BFloat16 instructions only. - 3, Both AArch64 Advanced SIMD and FP and AArch32 Advanced SIMD and VFP BFloat16 instructions.
advsimd_i8mm_support_level	int	0x0	Implement Int8 matrix multiply operations from ARMv8.6. AArch64 Advanced SIMD and FP Int8 matrix multiply instructions are automatically enabled when has_arm_v8-6 is true. - 0, Not implemented. - 1, AArch64 Advanced SIMD and FP Int8 matrix multiply instructions only. - 2, AArch32 Advanced SIMD and VFP Int8 matrix multiply instructions only. - 3, Both AArch64 Advanced SIMD and FP and AArch32 Advanced SIMD and VFP Int8 matrix multiply instructions.
amair_reg_rw_mask	int	0x0	RW mask for implementation-defined registers.
amu_aux_voffset_mask	int	0x0	If ARMv8.6 is implemented, each bit of the field, 0 to 15, when 1 indicates that the corresponding virtual offset register, AMEVCNTVOFF1<n>_EL2, is implemented
amu_has_external_interface	int	0x0	Implement external memory-mapped access to system register of activity monitor unit from ARMv8.4. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
amu_has_sysreg_interface	bool	0x1	Implement system register access to activity monitor unit from ARMv8.4.
amu_reset_domain	int	0x0	Reset domain for activity monitor unit. 0, COLD_RESET. 1, WARM_RESET. 2, NONE.
amu_version	int	0x1	Selects the activity monitor version implemented - 1, AMUv1 for Armv8.4 is implemented. - 2, AMUv1 for Armv8.6 is implemented.
apsr_read_restrict	bool	0x0	At EL0, unknown bits of APSR are RAZ.
arm_v8_7_accelerator_support_level	int	0x0	Implements Armv8.7 accelerator support instructions LD/ST64B, ST64BV, ST64BV0, etc.
auxilliary_feature_register0	int	0x0	Value of AFR0 ID register.
cache-log2linelen	int	0x6	Log2 of the cache line length in bytes.
cache_maintenance_hits_watchpoints	bool	0x0	DCIMVA operations executed in AArch32 modes hit watchpoints.
changing_block_size_without_bbm_support	int	0x0	Level of support for changing block size without break-before-make.
clear_IT_when_IL_set	bool	0x0	Clear IT bits when performing a *legal* exception return to AArch32 when IL is set

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-73 Parameters for Cluster_ARM_AEM-A_MP (continued)

Name	Type	Default value	Description
clear_IT_when_IL_set_explicitly	bool	0x0	Apart from clear_IT_when_IL_set, also clear IT bits when loading CPSR from SPSR/memory and IL == 1 in the value being loaded.
clear_reg_top_eret	int	0x1	Behaviour of the upper 32-bits of the Xn registers when changing between AArch32 state and AArch64 state. 0, upper 32-bits preserved for all registers. 1, upper 32-bits set to 0 for all accessible registers. 2, upper 32-bits set to 0 for a random selection of accessible registers. 3, upper-32-bits set to 0 for registers touched in AArch32
cpi_div	int	0x1	Divider for calculating CPI (Cycles Per Instruction)
cpi_mul	int	0x1	Multiplier for calculating CPI (Cycles Per Instruction)
dc_fault_unaligned_s1_device_s2_fwb	bool	0x0	Whether takes an Alignment Fault caused by the memory type on a DC {ZVA,GZVA,GVA} if the stage 1 memory type is any Device memory type
dcache-hit_latency	int	0x0	L1 D-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
dcache-maintenance_latency	int	0x0	L1 D-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
dcache-miss_latency	int	0x0	L1 D-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
dcache-prefetch_enabled	bool	0x0	Enable simulation of data cache prefetching. This is only used when dcache-state_modelled=true
dcache-read_access_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per access (of size dcache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
dcache-read_bus_width_in_bytes	int	0x8	L1 D-Cache read bus width in bytes used to calculate per-access timing annotations
dcache-read_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per byte accessed.dcache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
dcache-size	int	0x8000	L1 D-Cache size in bytes.
dcache-snoop_data_transfer_latency	int	0x0	L1 D-Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-73 Parameters for Cluster_ARM_AEM-A_MP (continued)

Name	Type	Default value	Description
dcache-ways	int	0x2	L1 D-Cache number of ways (sets are implicit from size).
dcache-write_access_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per access (of size dcache-write_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-write_latency is set. This is only used when dcache-state_modelled=true.
dcache-write_bus_width_in_bytes	int	0x8	L1 D-Cache write bus width in bytes used to calculate per-access timing annotations
dcache-write_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per byte accessed. dcache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
dcimva_requires_s2_write_permissions	bool	0x0	Data-cache invalidate by MVA operations require stage 2 write permission (virtualised AArch32 guest).
dczva_reports_lowest_addr_on_tag_check_fail	bool	0x0	Whether DC ZVA reports lowest address in FAR on tag check fail
delayed_dbgreg_between_secure_views	int	0x1	If delayed_dbgreg is enabled, whether the secure and nonsecure external views require explicit synchronization. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
dic-spi_count	int	0x40	Number of shared peripheral interrupts implemented
disable_sve_plugin	bool	0x0	If true, SVE will not be implemented in this processor even if the plugin is loaded
disable_unknown_update_event_on_reset	bool	0x0	Disables SYSREG_UPDATE event notification on reset for the registers whose bitfields are all reserverd or resets to architecturally unknown value
ecv_support_level	int	0x0	Implement Enhanced Counter Virtualization feature from ARMv8.6. 0, Not supported. 1, fully supported without CNTPOFF. 2, fully supported with CNTPOFF.
el0_can_access_imp_def_functionality	bool	0x0	If not made UNDEF by imp_def_functionality_behaviour, EL0 can access IMPLEMENTATION DEFINED registers and system instructions.
el0_el1_only_non_secure	bool	0x0	Secure/non-secure state if EL2 and EL3 are not implemented. 0, secure. 1, non-secure.
enable_tlb_contig_check	bool	0x0	Perform extra pagetable walks to check translation table entries that have the contiguous hint bit set.
enhanced_pac2_level	int	0x0	Implements Enhanced PAC2 from ARMv8.6. This feature is mandatory for ARMv8.6 but can be cherrypicked to a ARMv8.3(or greater) implementation. 0: No EnhancedPAC2, 1: EnhancedPAC2 Only, 2: EnhancedPAC2 with FPAC, 3: EnhancedPAC2 with FPACCombined.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-73 Parameters for Cluster_ARM_AEM-A_MP (continued)

Name	Type	Default value	Description
error_record_feature_register	string	""	RAS feature register values. An array of JSON objects. The JSON schema for the array is: [{"ED":0x0,"IMPDEF_3_2":0x0,"UI":0x0,"FI":0x0,"UE":0x0,"CFI":0x0,"CEC":0x0,"RP":0x0,"DUI":0x0,"CEO":0x0,"CI":0x0,"TS":0x0,"INJ":0x0,"FRX":0x0,"UC":0x0,"UEU":0x0,"UER":0x0,"UEO":0x0,"DE":0x0,"CE":0x0,"Visibility":"Core"},other_feature_register_values]. Where ED,UI,FI,CE and UE have valid values betwn 0x0 - 0x3. CFI and DUI have valid values 0x0, 0x2 and 0x3. CEC has valid values 0x0,0x2 or 0x4. RP,CEO,INJ,FRX,UC,UEU,UER,UEO,DE has valid values 0x0 or 0x1. CI and TS has valid values of 0x0, 0x1 and 0x2. Visibility has valid values "Core" or "Cluster"
error_record_feature_register_json_file	string	""	File path to the RAS feature register values as JSON. The file uses the same format as the error_record_feature_register parameter value
exercise_stxr_fail	bool	0x0	Reject a pseudo-random majority of exclusive store instructions
ext_abort_device_GRE_prefetch_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_index, Valid indices in range [0, number_of_error_records).
ext_abort_device_GRE_prefetch_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_device_GRE_read_is_critical	bool	0x0	Critical reporting of device-GRE read external aborts.
ext_abort_device_GRE_read_is_sync	int	0x2	Synchronous reporting of device-GRE read external aborts. 0, asynchronous. 1, synchronous. 2, same as ext_abort_device_read_is_sync.
ext_abort_device_GRE_read_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_device_read_ras_index, Valid indices in range [0, number_of_error_records).
ext_abort_device_GRE_read_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_device_read_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_device_GRE_write_is_critical	bool	0x0	Critical reporting of device-GRE write external aborts
ext_abort_device_GRE_write_is_sync	int	0x2	Synchronous reporting of device-GRE write external aborts. 0, asynchronous. 1, synchronous. 2, same as ext_abort_device_write_is_sync.
ext_abort_device_GRE_write_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_device_write_ras_index, Valid indices in range [0, number_of_error_records).
ext_abort_device_GRE_write_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_device_write_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-73 Parameters for Cluster_ARM_AEM-A_MP (continued)

Name	Type	Default value	Description
ext_abort_device_nGRE_prefetch_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_index, Valid indices in range [0, number_of_error_records).
ext_abort_device_nGRE_prefetch_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_device_nGRE_read_is_critical	bool	0x0	Critical reporting of device-nGRE read external aborts.
ext_abort_device_nGRE_read_is_sync	int	0x2	Synchronous reporting of device-nGRE read external aborts. 0, asynchronous. 1, synchronous. 2, same as ext_abort_device_read_is_sync.
ext_abort_device_nGRE_read_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_device_read_ras_index, Valid indices in range [0, number_of_error_records).
ext_abort_device_nGRE_read_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_device_read_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_device_nGRE_write_is_critical	bool	0x0	Critical reporting of device-nGRE write external aborts
ext_abort_device_nGRE_write_is_sync	int	0x2	Synchronous reporting of device-nGRE write external aborts. 0, asynchronous. 1, synchronous. 2, same as ext_abort_device_write_is_sync.
ext_abort_device_nGRE_write_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_device_write_ras_index, Valid indices in range [0, number_of_error_records).
ext_abort_device_nGRE_write_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_device_write_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_device_prefetch_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_index, Valid indices in range [0, number_of_error_records).
ext_abort_device_prefetch_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_device_read_is_critical	bool	0x0	Critical reporting of device-nGnRE read external aborts
ext_abort_device_read_is_sync	bool	0x1	Synchronous reporting of device-nGnRE read external aborts
ext_abort_device_read_ras_index	int	0x0	External Aborts are reported in RAS record index specified in this param. Values: Valid indices in range [0, number_of_error_records).
ext_abort_device_read_ras_type	int	0x0	External Aborts are reported as RAS error type specified in this param. Values: 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-73 Parameters for Cluster_ARM_AEM-A_MP (continued)

Name	Type	Default value	Description
ext_abort_device_write_is_critical	bool	0x0	Critical reporting of device-nGnRE write external aborts
ext_abort_device_write_is_sync	bool	0x0	Synchronous reporting of device-nGnRE write external aborts
ext_abort_device_write_ras_index	int	0x0	External Aborts are reported in RAS record index specified in this param. Values: Valid indices in range [0, number_of_error_records).
ext_abort_device_write_ras_type	int	0x0	External Aborts are reported as RAS error type specified in this param. Values: 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_fill_data	int	-0x202020303020203	Returned data, if external aborts are asynchronous
ext_abort_normal_cacheable_read_is_critical	bool	0x0	Critical reporting of normal write-back cacheable-read external aborts
ext_abort_normal_cacheable_read_is_sync	bool	0x1	Synchronous reporting of normal write-back cacheable-read external aborts
ext_abort_normal_cacheable_read_ras_index	int	0x0	External Aborts are reported in RAS record index specified in this param. Values: Valid indices in range [0, number_of_error_records).
ext_abort_normal_cacheable_read_ras_type	int	0x0	External Aborts are reported as RAS error type specified in this param. Values: 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_normal_cacheable_write_is_critical	bool	0x0	Critical reporting of normal write-back cacheable write external aborts
ext_abort_normal_cacheable_write_is_sync	bool	0x0	Synchronous reporting of normal write-back cacheable write external aborts
ext_abort_normal_cacheable_write_ras_index	int	0x0	External Aborts are reported in RAS record index specified in this param. Values: Valid indices in range [0, number_of_error_records).
ext_abort_normal_cacheable_write_ras_type	int	0x0	External Aborts are reported as RAS error type specified in this param. Values: 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_normal_noncacheable_prefetch_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_index, Valid indices in range [0, number_of_error_records).
ext_abort_normal_noncacheable_prefetch_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_normal_noncacheable_read_is_critical	bool	0x0	Critical reporting of normal noncacheable-read external aborts
ext_abort_normal_noncacheable_read_is_sync	bool	0x1	Synchronous reporting of normal noncacheable-read external aborts

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-73 Parameters for Cluster_ARM_AEM-A_MP (continued)

Name	Type	Default value	Description
ext_abort_normal_noncacheable_read_ras_index	int	0x0	External Aborts are reported in RAS record index specified in this param. Values: Valid indices in range [0, number_of_error_records).
ext_abort_normal_noncacheable_read_ras_type	int	0x0	External Aborts are reported as RAS error type specified in this param. Values: 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_normal_noncacheable_write_is_critical	bool	0x0	Critical reporting of normal noncacheable write external aborts
ext_abort_normal_noncacheable_write_is_sync	bool	0x0	Synchronous reporting of normal noncacheable write external aborts
ext_abort_normal_noncacheable_write_ras_index	int	0x0	External Aborts are reported in RAS record index specified in this param. Values: Valid indices in range [0, number_of_error_records).
ext_abort_normal_noncacheable_write_ras_type	int	0x0	External Aborts are reported as RAS error type specified in this param. Values: 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_normal_wt_cacheable_prefetch_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_index, Valid indices in range [0, number_of_error_records).
ext_abort_normal_wt_cacheable_prefetch_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_normal_wt_cacheable_read_is_critical	bool	0x0	Critical reporting of normal write-through cacheable-read external aborts
ext_abort_normal_wt_cacheable_read_is_sync	int	0x2	Synchronous reporting of normal write-through read external aborts. 0, asynchronous. 1, synchronous. 2, same as ext_abort_normal_cacheable_read_is_sync.
ext_abort_normal_wt_cacheable_read_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_normal_cacheable_read_ras_index, Valid indices in range [0, number_of_error_records).
ext_abort_normal_wt_cacheable_read_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_normal_cacheable_read_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_normal_wt_cacheable_write_is_critical	bool	0x0	Critical reporting of normal write-through write external aborts
ext_abort_normal_wt_cacheable_write_is_sync	int	0x2	Synchronous reporting of normal write-through write external aborts. 0, asynchronous. 1, synchronous. 2, same as ext_abort_normal_cacheable_write_is_sync.
ext_abort_normal_wt_cacheable_write_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_normal_cacheable_write_ras_index, Valid indices in range [0, number_of_error_records).

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-73 Parameters for Cluster_ARM_AEM-A_MP (continued)

Name	Type	Default value	Description
ext_abort_normal_wt_cacheable_write_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_normal_cacheable_write_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_prefetch_device_GRE_read_is_critical	bool	0x0	Critical reporting of external aborts generated by device-GRE instruction fetches
ext_abort_prefetch_device_GRE_read_is_sync	int	0x2	Behaviour of external aborts generated by device-GRE instruction fetches. 0, asynchronous abort. 1, synchronous abort. 2, same as ext_abort_prefetch_is_sync.
ext_abort_prefetch_device_nGRE_read_is_critical	bool	0x0	Critical reporting of external aborts generated by device-nGRE instruction fetches
ext_abort_prefetch_device_nGRE_read_is_sync	int	0x2	Behaviour of external aborts generated by device-nGRE instruction fetches. 0, asynchronous abort. 1, synchronous abort. 2, same as ext_abort_prefetch_is_sync.
ext_abort_prefetch_device_read_is_critical	bool	0x0	Critical reporting of external aborts generated by device-nGnRE instruction fetches
ext_abort_prefetch_device_read_is_sync	int	0x2	Behaviour of external aborts generated by device-nGnRE instruction fetches. 0, asynchronous abort. 1, synchronous abort. 2, same as ext_abort_prefetch_is_sync.
ext_abort_prefetch_is_critical	bool	0x0	Critical reporting of external aborts generated by normal writeback cacheable instruction fetches
ext_abort_prefetch_is_sync	bool	0x1	Behaviour of external aborts generated by normal writeback cacheable instruction fetches. 0, asynchronous abort. 1, synchronous abort.
ext_abort_prefetch_noncacheable_read_is_critical	bool	0x0	Critical reporting of external aborts generated by normal noncacheable instruction fetches
ext_abort_prefetch_noncacheable_read_is_sync	int	0x2	Behaviour of external aborts generated by normal noncacheable instruction fetches. 0, asynchronous abort. 1, synchronous abort. 2, same as ext_abort_prefetch_is_sync.
ext_abort_prefetch_ras_index	int	0x0	External Aborts are reported in RAS record index specified in this param. Values: Valid indices in range [0, number_of_error_records).
ext_abort_prefetch_ras_type	int	0x0	External Aborts are reported as RAS error type specified in this param. Values: 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_prefetch_so_read_is_critical	bool	0x0	Critical reporting of external aborts generated by device-nGnRnE instruction fetches
ext_abort_prefetch_so_read_is_sync	int	0x2	Behaviour of external aborts generated by device=nGnRnE instruction fetches. 0, asynchronous abort. 1, synchronous abort. 2, same as ext_abort_prefetch_is_sync.
ext_abort_prefetch_wt_cacheable_read_is_critical	bool	0x0	Critical reporting of external aborts generated by normal writethrough cacheable instruction fetches

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-73 Parameters for Cluster_ARM_AEM-A_MP (continued)

Name	Type	Default value	Description
ext_abort_prefetch_wt_cacheable_read_is_sync	int	0x2	Behaviour of external aborts generated by normal writethrough cacheable instruction fetches. 0, asynchronous abort. 1, synchronous abort. 2, same as ext_abort_prefetch_is_sync.
ext_abort_so_prefetch_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_index, Valid indices in range [0, number_of_error_records).
ext_abort_so_prefetch_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_so_read_is_critical	bool	0x0	Critical reporting of device-nGnRnE read external aborts
ext_abort_so_read_is_sync	bool	0x1	Synchronous reporting of device-nGnRnE read external aborts
ext_abort_so_read_ras_index	int	0x0	External Aborts are reported in RAS record index specified in this param. Values: Valid indices in range [0, number_of_error_records).
ext_abort_so_read_ras_type	int	0x0	External Aborts are reported as RAS error type specified in this param. Values: 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_so_write_is_critical	bool	0x0	Critical reporting of device-nGnRnE write external aborts
ext_abort_so_write_is_sync	bool	0x1	Synchronous reporting of device-nGnRnE write external aborts
ext_abort_so_write_ras_index	int	0x0	External Aborts are reported in RAS record index specified in this param. Values: Valid indices in range [0, number_of_error_records).
ext_abort_so_write_ras_type	int	0x0	External Aborts are reported as RAS error type specified in this param. Values: 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_ttw_cacheable_read_is_critical	bool	0x0	Critical reporting of TTW cacheable read external aborts
ext_abort_ttw_cacheable_read_is_sync	bool	0x1	Synchronous reporting of TTW cacheable read external aborts
ext_abort_ttw_cacheable_read_ras_index	int	0x0	External Aborts are reported in RAS record index specified in this param. Values: Valid indices in range [0, number_of_error_records).
ext_abort_ttw_cacheable_read_ras_type	int	0x0	External Aborts are reported as RAS error type specified in this param. Values: 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_ttw_noncacheable_read_is_critical	bool	0x0	Critical reporting of TTW noncacheable read external aborts
ext_abort_ttw_noncacheable_read_is_sync	bool	0x1	Synchronous reporting of TTW noncacheable read external aborts
ext_abort_ttw_noncacheable_read_ras_index	int	0x0	External Aborts are reported in RAS record index specified in this param. Values: Valid indices in range [0, number_of_error_records).

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-73 Parameters for Cluster_ARM_AEM-A_MP (continued)

Name	Type	Default value	Description
ext_abort_ttw_noncacheable_read_ras_type	int	0x0	External Aborts are reported as RAS error type specified in this param. Values: 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_ttw_wt_cacheable_read_is_critical	bool	0x0	Critical reporting of TTW write-through cacheable read external aborts
ext_abort_ttw_wt_cacheable_read_is_sync	int	0x2	Synchronous reporting of TTW write-through cacheable read external aborts. 0, asynchronous. 1, synchronous. 2, same as ext_abort_ttw_cacheable_read_is_sync.
ext_abort_ttw_wt_cacheable_read_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_ttw_cacheable_read_ras_index, Valid indices in range [0, number_of_error_records).
ext_abort_ttw_wt_cacheable_read_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_ttw_cacheable_read_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
fault_on_nT_bit_set	bool	0x1	Whether block translation table entries with the nT bit set should always fault. Only applies when changing_block_size_without_bbm_support_level is 1 or higher.
fault_unalign_to_unsupported_mem	int	0x0	If has_unaligned_single_copy_atomicity is true, whether unaligned A64 atomic, exclusive and acquire/release instructions to non iWB-oWB generate fault. This parameter is a bitmap. Bits 0,1,2 should be set accordingly to enable the fault behaviour. bit 0(LSB): atomic access should fault, bit 1: exclusive access should fault, bit 2: acquire/release should fault.
fault_unaligned_s1_device_s2_fwb	bool	0x0	Whether unaligned access with stage1 Device memory and final memory attribute forced to normal by FWB can generate alignment fault
force_deterministic_irg_tag_generation	bool	0x0	Force the random tag generated by the IRG instruction when GCR_EL1.RRND=1 to equal RGSR_EL1.SEED[3:0] rather than a non-deterministic value
fsr_ext_bit_update_kind	int	0x3	Set/Clear DFSR/IFSR EA bit on Synchronous/Async External Aborts. 0: Never Set, 1: Set on Synchronous Ext Aborts 2: Set on Asynchronous Ext Aborts 3: Set on both Sync and Async Ext Aborts.
global_debug_rom.ROMDEVID	int	0x0	Value of Debug Rom Device Identification Register
global_debug_rom.ROMPIDR	int	0x4000bb000	Value of Debug Rom Peripheral Identification Register
hardware_translation_table_update_implemented	int	0x1	Implement hardware translation table updates from ARMv8.1. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.1 is enabled. - 2, feature is implemented.
has_16bit_asids	bool	0x1	Enable 16-bit ASIDs.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-73 Parameters for Cluster_ARM_AEM-A_MP (continued)

Name	Type	Default value	Description
has_16bit_vmids	int	0x1	Implement support for 16-bit VMIDs from ARMv8.1. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.1 is enabled. - 2, feature is implemented.
has_16k_granule	bool	0x0	Implement the 16k LPAE translation granule.
has_4k_granule	bool	0x1	Implement the 4k LPAE translation granule.
has_52bit_address_with_16k	int	0x0	Implements Armv8.7 52-bit IPA/PA support for 16k Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if Armv8.7 is enabled. - 2, feature is implemented.
has_52bit_address_with_4k	int	0x0	Implements Armv8.7 52-bit IPA/PA support for 4k Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if Armv8.7 is enabled. - 2, feature is implemented.
has_64k_granule	bool	0x1	Implement the 64k LPAE translation granule.
has_aarch32_hpd	bool	0x0	If true then hierarchical permission disable is supported in AArch32
has_aarch64	bool	0x1	All implemented exception levels can run in AArch64
has_actlr2	bool	0x0	If true ACLTR2 exists and ACTLR2(NS) is aliased to ACTLR_EL1[63:32]
has_amu	int	0x0	Implement activity monitor functionality from ARMv8.4. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_arm_v8-1	bool	0x0	Implement the ARMv8.1 Extension.
has_arm_v8-2	bool	0x0	Implement the ARMv8.2 Extension.
has_arm_v8-3	bool	0x0	Implement the ARMv8.3 Extension.
has_arm_v8-4	bool	0x0	Implement the ARMv8.4 Extension.
has_arm_v8-5	bool	0x0	Implement the ARMv8.5 Extension.
has_arm_v8-6	bool	0x0	Implement the ARMv8.6 Extension.
has_arm_v8-7	bool	0x0	Implement the Armv8.7 Extension. This feature is incomplete and under development.
has_at_with_pan	int	0x1	Implement new AT instructions with PAN support. Possible values of this parameter are: - 1, feature is implemented if ARMv8.2 is enabled. - 2, feature is implemented.
has_axflag_xaflag	int	0x1	Implement flag manipulation instructions (AXFlag, XAFlag) from ARMv8.5. Possible values of this parameter are: - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-73 Parameters for Cluster_ARM_AEM-A_MP (continued)

Name	Type	Default value	Description
has_axflag_xaflag_frint	int	0x1	Implement flag manipulation instructions (AXFlag, XAFlag) and floating-point rounding to int instructions (FRINT[32|64][X|Z]) from ARMv8.5. If this parameter is enabled, it also enables both has_axflag_xaflag and has_frint. If support for only one of the features is needed, please use the individual parameters and do not enable this one. Possible values of this parameter are: - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
has_branch_target_exception	int	0x0	Implement Branch target identification mechanism from ARMv8.5. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
has_ccidx	bool	0x0	Implement the ARMv8.3 CCSIDR Extension. Extending the ccsidr number of sets.
has_cfinv_rmif_setf	int	0x1	Implement flag manipulation (CFINV, RMIF, SETF8, SETF16) instructions from ARMv8.4. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_coherent_icache	bool	0x0	Whether icache invalidation to the point of unification is required for instruction to data coherence. true - Invalidate operations not required
has_common_not_private_translations	int	0x1	Implement the TTBRn_ELx.CnP (Common not Private) controls from ARMv8.2. Possible values of this parameter are: - 1, feature is implemented if ARMv8.2 is enabled. - 2, feature is implemented.
has_complex_number	int	0x1	Implement ARMv8.3 complex number support, Multiply Accumulate and Add instructions. Possible values of this parameter are: - 1, feature is implemented if ARMv8.3 is enabled. - 2, feature is implemented.
has_cvadp_support	int	0x0	Implement instruction to support cache clean by deep persistence (DC CVADP) from ARMv8.5, can be selected for core implemented on any arch version starting ARMv8.2 Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.2 is enabled. - 2, feature is implemented.
has_data_alignment_flag	int	0x0	Implement non-optimal misalignment flag for PMU/SPE from ARMv8.5 Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
has_delayed_ctireg	bool	0x0	Delay the functional effect of CTI register writes until ISB or implicit barrier.
has_delayed_dbgreg	bool	0x0	Delay the functional effect of external debug register writes until ISB or implicit barrier.
has_delayed_sysreg	bool	0x0	Delay the functional effect of system register writes until ISB or implicit barrier.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-73 Parameters for Cluster_ARM_AEM-A_MP (continued)

Name	Type	Default value	Description
has_delayed_wfe_trap	int	0x0	Implements Configurable Delayed WFE trapping from ARMv8.6 Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.6 is enabled. - 2, feature is implemented.
has_dgh	int	0x1	Implements Data Gathering Hint instruction from ARMv8.6 Possible values of this parameter are: - 1, feature is implemented if ARMv8.6 is enabled. - 2, feature is implemented.
has_dot_product	int	0x1	Implement the dot product (UDOT, SDOT) instructions from ARMv8.4. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_e0pd	int	0x1	Implement ARMv8.5 feature to prevent unprivileged access to one half of the memory Possible values of this parameter are: - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
has_el2	bool	0x1	Implements EL2
has_el3	bool	0x1	Implements EL3
has_enhanced_pac	bool	0x0	If pointer authentication is enabled then implement enhanced PAC.
has_enhanced_pan	int	0x1	Implements Armv8.7 Enhanced PAN feature Possible values of this parameter are: - 1, feature is implemented if Armv8.7 is enabled. - 2, feature is implemented.
has_ets	bool	0x0	Whether Enhanced Translation Synchronization is supported
has_extended_recp_estimate	int	0x0	Implements increased precision of reciprocal instructions.) Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if Armv8.7 is enabled. - 2, feature is implemented.
has_fgt	int	0x1	Implements Fine-grained Virtualization Traps extension from ARMv8.6 Possible values of this parameter are: - 1, feature is implemented if ARMv8.6 is enabled. - 2, feature is implemented.
has_fp16	int	0x1	Implement the half-precision floating-point data processing instructions from ARMv8.2. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.2 is enabled. - 2, feature is implemented.
has_fp16_fmlal	int	0x1	Implement the New Floating Point Multiplication Variant (FP16 FMLAL, FMLSL) instructions from ARMv8.4. Only supported if has_fp16=0x1. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_frint	int	0x1	Implement floating-point rounding to int instructions (FRINT[32|64][X|Z]) from ARMv8.5. Possible values of this parameter are: - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
has_generic_authentication	int	0x1	Implement ARMv8.3 generic authentication. Possible values of this parameter are: - 1, feature is implemented if ARMv8.3 is enabled. - 2, feature is implemented.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-73 Parameters for Cluster_ARM_AEM-A_MP (continued)

Name	Type	Default value	Description
has_guest_translation_granule	int	0x1	Implement mechanism for guest translation granule identification from ARMv8.5, ID values determined by stage1 granule configuration parameters Possible values of this parameter are: - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
has_hardware_translation_table_update	int	0x2	Type of hardware translation table supported (when enabled by hardware_translation_table_update_implemented). 0, not implemented. 1, access bit updates implemented. 2, access bit updates and dirty bit mechanism implemented.
has_hcrx_el2	int	0x1	Implements new HCRX_EL2 id register from Armv8.7 Possible values of this parameter are: - 1, feature is implemented if Armv8.7 is enabled. - 2, feature is implemented.
has_id_reg_read	int	0x1	Implement read access to the ID registers (ESR_ELx.EC=0x18) Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_iesb	int	0x1	Implement support for implicit error sync event from ARMv8.2. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.2 is enabled. - 2, feature is implemented.
has_jscvt	int	0x1	Implement ARMv8.3 javascript Floating-point to Integer conversion instruction. Possible values of this parameter are: - 1, feature is implemented if ARMv8.3 is enabled. - 2, feature is implemented.
has_large_system_ext	bool	0x0	Implement the ARMv8 Large System Extensions.
has_large_va	int	0x0	Implement support for the extended 52-bit virtual addresses from ARMv8.2. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.2 is enabled. - 2, feature is implemented.
has_ldapur_stlur	int	0x1	Implement support for LDAPR and STLR instructions with immediate offsets from ARMv8.4. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_ldm_stm_ordering_control	int	0x0	Implement the SCTLR_ELx.LSMAOE (Load/Store Multiple Atomicity and Ordering Enable) and SCTLR_ELx,nTLSMD (no Trap Load/Store Multiple to Device) controls from ARMv8.2. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.2 is enabled. - 2, feature is implemented.
has_lrcpc	bool	0x0	If true then it support the RCpc feature (ARMv8.3)
has_mpam	int	0x0	Implement ARMv8.4 MPAM Registers and associated functionality. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-73 Parameters for Cluster_ARM_AEM-A_MP (continued)

Name	Type	Default value	Description
has_mt_pmu_disable_feature	int	0x0	Implements Multi-threading PMU disable extension from ARMv8.6 Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.6 is enabled. - 2, feature is implemented.
has_mte_ctrl_bits_stateful	bool	0x0	if memory_tagging_support_level == 1, Whether mte specific control bits in system registers are stateful
has_nested_virtualization	int	0x1	Implement ARMv8.3 nested virtualization. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.3 is enabled. - 2, feature is implemented.
has_no_os_double_lock	int	0x0	Do not implement the OS double-lock. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_non_context_synchronizing_exception_controls	int	0x1	Implement cosmetic controls for whether exception entry and exit are context synchronizing events (SCTLR_ELx.{EIS,EOS}) from ARMv8.5. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
has_par_bit10_razwi	bool	0x0	Whether PAR_EL1[10] is RAZ/WI
has_pointer_authentication	int	0x1	Implement ARMv8.3 pointer authentication. Possible values of this parameter are: - 1, feature is implemented if ARMv8.3 is enabled. - 2, feature is implemented.
has_prediction_invalidation_instructions	int	0x1	Implement execution and data prediction invalidation from ARMv8.5 Possible values of this parameter are: - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
has_pstate_dit	int	0x1	Implement timing insensitivity of data processing instructions from ARMv8.4. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_pstate_pan	int	0x1	Implement the PSTATE.PAN (Privileged Access Never) control from ARMv8.1 Possible values of this parameter are: - 1, feature is implemented if ARMv8.1 is enabled. - 2, feature is implemented.
has_pstate_uao	int	0x1	Implement the PSTATE.UAO (User Access Override) control from ARMv8.2. Possible values of this parameter are: - 1, feature is implemented if ARMv8.2 is enabled. - 2, feature is implemented.
has_ras_armv84_extension	int	0x1	Implement ARMv8.4 RAS Extension. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_ras_critical_error	int	0x0	[DEPRECATED: Set CI field on first register in error_record_feature_register JSON instead] ARMv8.4 AArch64 RAS Critical Error is implemented or not. 0 - Feature Not Supported, 1 - Feature always enabled, 2 - Feature is controllable.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-73 Parameters for Cluster_ARM_AEM-A_MP (continued)

Name	Type	Default value	Description
has_ras_double_fault	int	0x1	Implement ARMv8.4 RAS Doublt Fault Extension. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_ras_fault_injection	int	0x0	[DEPRECATED: Set INJ field on first register in error_record_feature_register JSON instead] Implement ARMv8.4 Standard Fault Injection mechanism. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_ras_timestamp	int	0x0	[DEPRECATED: Set TS field on first register in error_record_feature_register JSON instead] ARMv8.4 AArch64 RAS Timestamp register is implemented or not. 0 - No Timestamp is recorded, 1 - Generic Timer timestamp is recorded, 2 - IMP DEF timestamp is recorded.
has_restriction_on_speculative_data_loaded	int	0x1	Implements the ARMv8.5 security feature (Restrictions on the effects of speculation) Possible values of this parameter are: - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
has_rndr	int	0x0	Implement random number instructions to read from RNDR and RNDRSS random number registers from ARMv8.5. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
has_rounding_doubling_multiply_add_subtract	int	0x1	Implement the rounding doubling multiply add and subtract instructions from ARMv8.1 Possible values of this parameter are: - 1, feature is implemented if ARMv8.1 is enabled. - 2, feature is implemented.
has_secure_el2	int	0x1	Implement support for Secure EL2 Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_self_hosted_trace_extension	int	0x1	Implement support for the Self-hosted Trace Extensions from ARMv8.4. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_small_page_table	int	0x1	Implement small page table support which increases the maximum value of TxSZ field from ARMv8.4. Note: will be unimplemented only if both has_small_page_table=0x0 and has_secure_el2=0x0. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_speculation_barrier_inst	int	0x1	Implement speculation barrier instruction (SB) from ARMv8.5 Possible values of this parameter are: - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
has_stage2_fwb	int	0x1	Implement HCR_EL2.FWB, stage 2 control of memory types and cacheability Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-73 Parameters for Cluster_ARM_AEM-A_MP (continued)

Name	Type	Default value	Description
has_stage2_xnx	int	0x1	Implement the extended XN[1:0] stage 2 control from ARMv8.2. Possible values of this parameter are: - 1, feature is implemented if ARMv8.2 is enabled. - 2, feature is implemented.
has_stage2_xnx_in_aarch32	bool	0x1	Implement the extended XN[1:0] stage 2 control from ARMv8.2 for Aarch32.
has_statistical_profiling	bool	0x1	Whether Statistical Based Profiling is implemented
has_supersections	bool	0x1	Whether VMSAv8-32 supersection to support more than 32-bit PA using short descriptor is implemented.
has_tlb_conflict_abort	bool	0x0	Detected inconsistent TLB content generate aborts.
has_tlbi_range	int	0x1	Implement support for TLB Range Maintenance instructions (TLBI RVAE1, etc) from ARMv8.4. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_tlbi_to_outer_shareable	int	0x1	Implement support for TLB Maintenance instructions that extend to the Outer Shareable domain (TLBI VAE1OS, etc) from ARMv8.4. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_tlbi_ttl	int	0x1	Implement support for the TTL level hint in by-address TLB Maintenance instructions from ARMv8.4. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_unaligned_single_copy_atomicity	int	0x1	Implement support for SCTLR_ELx.nAA from ARMv8.4, and A64 atomic, exclusive and acquire/release instructions accessing unaligned bytes inside a 16byte window will not generate alignment fault. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_v8_4_debug_extension	int	0x1	Implement ARMv8.4 debug extensions Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_v8_4_pmu_extension	int	0x1	Implement PMU extension from ARMv8.4. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_v8_5_debug_over_power_down	int	0x0	Implement ARMv8.5 Debug over powerdown Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
has_v8_5_pmu_extension	int	0x1	Implement PMU extension from ARMv8.5. Possible values of this parameter are: - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
has_v8_5_spe_extension	int	0x1	Implement SPE extension from ARMv8.5 Possible values of this parameter are: - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-73 Parameters for Cluster_ARM_AEM-A_MP (continued)

Name	Type	Default value	Description
has_v8_6_pmu_events	int	0x1	Implements PMU events from ARMv8.6 Possible values of this parameter are: - 1, feature is implemented if ARMv8.6 is enabled. - 2, feature is implemented.
has_v8_7_fp_enhancements	int	0x1	Implements the Floating Point enhancements from Armv8.7 (introduces FPCR.FIZ/AH/NEP, etc.) Possible values of this parameter are: - 1, feature is implemented if Armv8.7 is enabled. - 2, feature is implemented.
has_v8_7_pmu_events	int	0x1	Implement PMU events from ARMv8.7. Possible values of this parameter are: - 1, feature is implemented if Armv8.7 is enabled. - 2, feature is implemented.
has_v8_7_pmu_extension	int	0x1	Implement PMU extension from ARMv8.7. Possible values of this parameter are: - 1, feature is implemented if Armv8.7 is enabled. - 2, feature is implemented.
has_v8_7_spe_extension	int	0x1	Implement SPE extension from ARMv8.7. Possible values of this parameter are: - 1, feature is implemented if Armv8.7 is enabled. - 2, feature is implemented.
has_vncr_el2	int	0x1	Implement support for nested virtualization enhancements from ARMv8.4. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_wfet_and_wfit	int	0x1	Implements WFE and WFI with Timeout from Armv8.7 Possible values of this parameter are: - 1, feature is implemented if Armv8.7 is enabled. - 2, feature is implemented.
has_writebuffer	bool	0x0	Implement write accesses buffering before L1 cache. May affect ext_abort behaviour.
has_xs	int	0x1	Implements Armv8.7 XS, TLBInXS, DSBnXS instruction Possible values of this parameter are: - 1, feature is implemented if Armv8.7 is enabled. - 2, feature is implemented.
icache-hit_latency	int	0x0	L1 I-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when icache-state_modelled=true.
icache-log2linelen	int	0x0	If nonzero, Log2 of the instruction cache line length in bytes (valid values in range 4-8). Otherwise the value of cache-log2linelen is used.
icache-maintenance_latency	int	0x0	L1 I-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when icache-state_modelled=true.
icache-miss_latency	int	0x0	L1 I-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when icache-state_modelled=true.
icache-nprefetch	int	0x1	Number of next sequential instruction cache lines to prefetch. This is only used when icache-prefetch_enabled=true.
icache-prefetch_enabled	bool	0x0	Enable simulation of instruction cache prefetching. This is only used when icache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-73 Parameters for Cluster_ARM_AEM-A_MP (continued)

Name	Type	Default value	Description
icache-prefetch_level	int	0x0	0 based cache level at which instructions are pre-fetched. This is only used when icache-prefetch_enabled=true.
icache-read_access_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per access (of size icache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if icache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when icache-state_modelled=true.
icache-read_bus_width_in_bytes	int	0x8	L1 I-Cache read bus width in bytes used to calculate per-access timing annotations
icache-read_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per byte accessed.icache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when icache-state_modelled=true.
icache-size	int	0x8000	L1 I-Cache size in bytes.
icache-ways	int	0x2	L1 I-Cache number of ways (sets are implicit from size).
ignore_DBGPRCR_CWRR	bool	0x0	Ignore writes to the deprecated DBGPRCR.CWRR bit.
ignore_access_flag_update_by_at_ops	bool	0x0	If true, AT operations do not update access flag
ignore_large_address_top_bits_in_page_walk	bool	0x0	Whether page table bits [15:12] are ignored if PA_SIZE < 52 and output address is configured < 52 with large page
imp_def_functionality_behaviour	int	0x0	Behaviour of IMPLEMENTATION DEFINED registers and system instructions. 0, UNDEF. 1, RAZ/WI.
impdef_regs_and_unpred_from_implementation	string	""	Configure implementation defined registers and unpredictable behaviour to match the specified implementation. Requires a license for the selected implementation model. Use ARM_Cortex-A<num> or ARM_<codename> for licensed pre-release cores.
independent_cache_control_traps	int	0x0	Implement Independent Cache Control traps from ARMv8.5. 0, NO_SUPPORT. 1, SUPPORTED_BUT_NOT_FOR_TLB_MAINTENANCE_INSTRUCTIONS. 2, FULL_SUPPORT.
insert_iesb_before_exception	bool	0x0	If true then inserts an IESB before taking with Exception otherwise has no effect and IESB is taken after PState is changed due to the Exception.
instruction_tlb_size	int	0x0	Number of stage1+2 itlb entries (or 0 for unified ITLB+DTLB)
is_mt_res0	bool	0x0	If ARMv8.6 is not implemented, and PMUv3 is implemented, this parameter controls whether PMEVTYPER<n>.MT bit is RES0 or RW. For other implementations, this parameter has no effect
is_uniprocessor	bool	0x0	Value for the U bit in MPIDR. true disables L1 cache coherency protocols

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-73 Parameters for Cluster_ARM_AEM-A_MP (continued)

Name	Type	Default value	Description
l2cache-hit_latency	int	0x0	L2 Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
l2cache-maintenance_latency	int	0x0	L2 Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
l2cache-miss_latency	int	0x0	L2 Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
l2cache-read_access_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
l2cache-read_bus_width_in_bytes	int	0x8	L2 Cache read bus width in bytes used to calculate per-access timing annotations
l2cache-read_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per byte accessed.l2cache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
l2cache-size	int	0x80000	L2 Cache size in bytes.
l2cache-snoop_data_transfer_latency	int	0x0	L2 Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
l2cache-snoop_issue_latency	int	0x0	L2 Cache timing annotation latency for snoop accesses issued by this cache in total ticks. This is only used when dcache-state_modelled=true.
l2cache-ways	int	0x10	L2 Cache number of ways (sets are implicit from size).
l2cache-write_access_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-write_latency is set. This is only used when dcache-state_modelled=true.
l2cache-write_bus_width_in_bytes	int	0x8	L2 Cache write bus width in bytes used to calculate per-access timing annotations
l2cache-write_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per byte accessed. l2cache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
l3cache-mpamf.bwa_width_ns	int	0x10	L3 Cache width of MPAM bandwidth allocation fields for non-secure accesses.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-73 Parameters for Cluster_ARM_AEM-A_MP (continued)

Name	Type	Default value	Description
l3cache-mpamf.bwa_width_s	int	0x10	L3 Cache width of MPAM bandwidth allocation fields for secure accesses.
l3cache-mpamf.cmax_width_ns	int	0x0	L3 Cache number of fractional bits in MPAM cache capacity partition control for non-secure accesses. Only the register interface is implemented - the control is NOT FUNCTIONAL.
l3cache-mpamf.cmax_width_s	int	0x0	L3 Cache number of fractional bits in MPAM cache capacity partition control for secure accesses. Only the register interface is implemented - the control is NOT FUNCTIONAL.
l3cache-mpamf.cpbm_width_ns	int	0x0	L3 Cache width of MPAM cache portion bitmap for non-secure accesses. If 0, the feature is not implemented, and all ways are available.
l3cache-mpamf.cpbm_width_s	int	0x0	L3 Cache width of MPAM cache portion bitmap for secure accesses. If 0, the feature is not implemented, and all ways are available.
l3cache-mpamf.csu_num_mon_ns	int	0x0	L3 Cache number of MPAM cache storage usage monitors for non-secure accesses.
l3cache-mpamf.csu_num_mon_s	int	0x0	L3 Cache number of MPAM cache storage usage monitors for secure accesses.
l3cache-mpamf.has_prop_ns	bool	0x0	Enable memory bandwidth proportional stride control for non-secure accesses. Only the register interface is implemented - the control is NOT FUNCTIONAL.
l3cache-mpamf.has_prop_s	bool	0x0	Enable memory bandwidth proportional stride control for secure accesses. Only the register interface is implemented - the control is NOT FUNCTIONAL.
max_32bit_el	int	0x3	Maximum exception level supporting AArch32 modes.
memory.acp.AxCACHE_mask	int	0x0	Used with memory.acp.AxCACHE_pattern to define which memory types the ACP port accepts. All transactions which do not satisfy (AxCACHE & mask) == pattern will abort.
memory.acp.AxCACHE_pattern	int	0x0	Used with memory.acp.AxCACHE_mask to define which memory types the ACP port accepts. All transactions which do not satisfy (AxCACHE & mask) == pattern will abort.
memory.l2_cache.is_inner_cacheable	bool	0x1	L2 cache obeys inner cacheable attributes (rather than outer cacheable attributes)
memory.l2_cache.is_inner_shareable	bool	0x1	L2 cache obeys inner shareable attributes (rather than outer sharaable attributes)
memory_tagging_support_level	int	0x0	Specify the memory tagging extension support level: 0, not implemented. 1, instructions and registers only are implemented. 2, implemented. 3, implemented with asymmetric handling of exceptions
mixed_endian	int	0x1	Implement support for mixed endianness. 0, not supported. 1, supported at all exception levels. 2, supported at EL0 only.
mpam_force_ns_rao	bool	0x0	Whether MPAM3_EL3.FORCE_NS bit is RAO/WI.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-73 Parameters for Cluster_ARM_AEM-A_MP (continued)

Name	Type	Default value	Description
mpam_frac	int	0x0	MPAM fractional revision number in ID_AA64PFR1_EL1.MPAM_frac field
mpam_has_hcr	bool	0x0	MPAM Whether MPAMIDR_EL1 HAS_HCR bit is set or clear
mpam_max_partid	int	0xffff	MPAM Maximum PARTID Supported
mpam_max_pmg	int	0xff	MPAM Maximum PMG Supported
mpam_max_vpmr	int	0x0	MPAM Maximum VPMR Supported
mpamidr_has_force_ns	int	0x0	Whether MPAMIDR_EL1.HAS_FORCE_NS bit is set or clear. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.6 is enabled. - 2, feature is implemented.
mpamidr_has_sdeflt	int	0x0	Whether MPAMIDR_EL1.HAS_SDEFLT bit is set or clear. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.6 is enabled. - 2, feature is implemented.
mpamidr_has_tidr	int	0x0	Whether MPAMIDR_EL1.HAS_TIDR bit is set or clear Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.6 is enabled. - 2, feature is implemented.
mvbar_reset_is_rvbar	bool	0x1	If true then the reset value of MVBAR is RVBAR, if false the reset value is UNKNOWN.
non_secure_vgic_alias_when_ns_only	int	0x0	If ! has_el3 and only non-secure side exists, then the normal position of the VGIC is a secure alias. If this parameter is non-zero then in addition a non-secure alias of the VGIC will be placed at this position (aligned to 32 KB).
num_loregion_descriptors	int	0x0	Number of Limited Ordering Region descriptors implemented (if ARM v8.1 extensions are implemented)
num_loregions	int	0x0	Number of Limited Ordering Regions implemented excluding background region (if ARM v8.1 extensions are implemented)
number_of_error_records	int	0x0	Cores Number of Error records supported for RAS
optimal_alignment_size	int	0x1	Alignment boundary which does not incur additional performance penalty from ARMv8.5. - 1, architectural misalignment is used to set PMU event LDST_ALIGN_LAT and SPE event E[11] - 2, access crossing 4 byte boundary is used to set PMU event LDST_ALIGN_LAT and SPE event E[11] - 3, access crossing 8 byte boundary is used to set PMU event LDST_ALIGN_LAT and SPE event E[11] ... - 12, access crossing 4 Kbyte boundary is used to set PMU event LDST_ALIGN_LAT and SPE event E[11]

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-73 Parameters for Cluster_ARM_AEM-A_MP (continued)

Name	Type	Default value	Description
output_attributes	string	"ExtendedID[62:55]=MPAM_PMG, ExtendedID[54:39]=MPAM_PARTID, ExtendedID[38]=MPAM_NS"	User-defined transform to be applied to bus attributes like MasterID, ExtendedID or UserFlags. Currently, only works for MPAM Attributes encoding into bus attributes.
page_based_hardware_attributes	int	0x0	Implement the page based hardware attributes from ARMv8.2. This parameter indicates which page table bits are available for hardware, where bits[3:0] correspond to PTE[62:59] and to TCR_ELx.HWUnyy.
pmmir_el1_bus_slots	int	0x0	Largest value by which BUS_ACCESS can increment over BUS_CYCLES cycles. From v8.7 PMU extension
pmmir_el1_bus_width	int	0x0	Width, in bytes, of accesses counted by BUS_ACCESS. From v8.7 PMU extension
pseudo_fault_generation_feature_register	string	""	ARMv8.4 Standard Pseudo-fault generation feature register values. JSON schema for the parameter value is: [{"OF":false,"UC":false,"UEU":false,"UER":false,"UEO":false,"DE":false,"CE":0x0,"CI":false,"ER":false,"PN":false,"AV":false,"MV":false,"SYN":false,"R":false},other_psuedo-fault_generating_features_register_values]. Where OF, UC, UEU, UER, UEO, DE, CI, ER, PN, AV, MV, SYN, and R have valid false(NOT_SUPPORTED) and true(FEATURE_CONTROLLABLE), where CE can have 0(NOT_SUPPORTED), 1(NONSPECIFIC_CE_SUPPORTED) and 3(TRANSIENT_OR_PERSISTENT_CE_SUPPORTED). Effective only when ERXFR's INJ field allows it or has_ras_fault_injection is true.
pstate_ssbs_type	int	0x0	Implement speculative store bypass safe feature from ARMv8.5. 0, Not supported. 1, Supported without MSR/MRS access to SSBS. 2, fully supported.
ptw_latency	int	0x0	Page table walker latency for TA (Timing Annotation), expressed in simulation ticks
randomize_unknowns_at_reset	bool	0x0	Will fill in unknown bits in registers at reset with random value using register_reset_data as seed, it overrides scramble_unknowns_at_reset
ras_errselr_undef_if_no_error_records	bool	0x0	If true, ERRSELR_EL1 will be undefined if ERRIDR_EL1 indicates that zero error records are implemented.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-73 Parameters for Cluster_ARM_AEM-A_MP (continued)

Name	Type	Default value	Description
ras_frac	int	0x0	0, No additional feature implemented. 1, Additional ERXMISC*, ERXPFG* registers and FaultInjection trap from RAS v1.1. implemented
register_reset_data	int	0x0	Data used to fill register bits when they become UNKNOWN at reset.
restriction_on_speculative_execution	int	0x0	Implements the ARMv8.5 security feature (Restrictions on the effects of speculation): 0: No disclosure whether branch targets trained in one context can affect speculative execution in a different context, 1: Branch targets trained in one context cannot affect speculative execution in a different hardware described context (SCXTNUM_ELx not supported), 2: Branch targets trained in one context cannot affect speculative execution in a different hardware described context (SCXTNUM_ELx supported)
rmr_always_implemented	bool	0x0	Always implement RMR_ELx, RMR, or HRMR at the highest implemented exception level, even if that exception level cannot use both AArch32 and AArch64.
rndr_rndrrs_seed	int	0x0	Initial seed for random engine used in RNDR register
scheduler_mode	int	0x0	Control the interleaving of instructions in this processor. 0, default long quantum. 1, low latency mode, short quantum and signal checking. 2, lock-breaking mode, long quantum with additional context switches near load-exclusive instructions.
scramble_unknowns_at_reset	bool	0x1	Will fill in unknown bits in registers at reset with register_reset_data
set_rasv10_for_armv84_and_higher	bool	0x0	ARMv8.4 mandates RAS System Architecture v1.1, but when there are no error records and ARMv8.4-DFE is not implemented then there is no functional difference between the RAS System Architecture v1.0 (that is, the RAS extension as in pre-ARMv8.4 implementations) and the RAS System Architecture v1.1 (also known as ARMv8.4-RAS). This flag if true will set the RAS ID to declare RAS v1.0 rather than RAS v1.1 for ARMv8.4 and higher implementations. If this is set and the core does not conform to the restrictions then this parameter is ignored.
spsr_el3_is_mapped_to_spsr_mon	bool	0x1	Whether SPSR_EL3 is mapped to AArch32 register SPSR_mon
spsr_m4_res0	bool	0x0	Whether SPSR_ELx.M[4] bit should be RES0 for AARCH64 only implementations
stage12_tlb_size	int	0x80	Number of stage1+2 tlb entries.
stage1_tlb_size	int	0x0	Number of stage1 only tlb entries.
stage1_walkcache_size	int	0x0	Number of stage1 only walk cache entries.
stage2_tlb_size	int	0x0	Number of stage2 only tlb entries.
stage2_walkcache_size	int	0x0	Number of stage2 only walk cache entries.
statistical_profiling_buffer_alignment	int	0x1	Statistical profiling alignment constraint for sample buffer

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-73 Parameters for Cluster_ARM_AEM-A_MP (continued)

Name	Type	Default value	Description
statistical_profiling_random_interval_is_separate	bool	0x0	Statistical profiling random interval gets added to the main timer interval(false) or (true) runs as separate timer
statistical_profiling_recommended_min_sampling	int	0x100	Statistical profiling recommended minimum sampling interval
take_ccfail_tsc_trap	bool	0x0	When take_ccfail_undef=1 this prameter controls whether or not an SMC instruction that is trapped by HCR_EL2.TSC but fails its condition code check generates a trap to EL2.
take_ccfail_undef	bool	0x1	UNDEF exception is taken even if condition code check fails
tcr_ps_reserved_value_size	int	0x0	Physical size treated when TCR.(I)PS is programmed with a reserved value. 0, 48 bits. 1, 52 bits. The parameter value is treated 0 if LPA is not supported.
tcr_txsz_undersize_should_fault	bool	0x1	If large VA is not supported, Whether undersized TxSZ value should generate translation fault
tdosa_traps_osdlr_if_no_os_double_lock	bool	0x1	MDCR_EL*.TDOSA enables trap on OSDLR_EL1 and DBGOSDLR when OS double-lock is not implemented.
tidcp_traps_el0_undef_imp_def	bool	0x1	TIDCP has priority over UNDEF for accesses to IMPLEMENTATION DEFINED functionality from EL0
tlb_latency	int	0x0	TLB latency for TA (Timing Annotation), expressed in simulation ticks
trace_physical_registers_when_host_virtualisation_enabled	int	0x0	When host virtualisation is enabled, trace sysreg accesses to physical register accessed (0=disabled, 1=Trace only ELR/SPSR_EL1 as ELR/SPSR_EL2, 2=Trace all redirected registers as physical registers
treat-dcache-cmos-to-pou-as-nop	bool	0x0	Whether dcache invalidation to the point of unification is required for instruction to data coherence. true - Invalidate operations not required
treat-dcache-invalidate-as-clean-invalidate	bool	0x0	Treat data cache invalidate operations as clean and invalidate.
undef_ccsidr2_access_for_unimplemented_aarch32	bool	0x0	Whether access to CCSIDR2 is undef if AArch32 is implemented or not
unification-level	int	0x1	Level of Unification Inner Shareable for the cache hierarchy
unification-uniprocessor-level	int	0x1	Level of Unification Uniprocessor for the cache hierarchy
unpred_nested_virtualization_nv_behaviour	int	0x0	Constrained unpredictable choices for HCR_EL2.NV=0 and HCR_EL2.NV1=1 with respect to nested virtualization - 0, Behave as defined in the sepcification as per bit values - 1, Behave as if HCR_EL2.NV=1 and HCR_EL2.NV1=1 for all purpose other than reading back HCR_EL2.NV - 2, Behave as if HCR_EL2.NV=0 and HCR_EL2.NV1=0 for all purpose other than reading back HCR_EL2.NV1

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-73 Parameters for Cluster_ARM_AEM-A_MP (continued)

Name	Type	Default value	Description
unpred_vncr_el2_ress_mismatch	int	0x0	Constrained unpredictable choices when bits marked as RESS do not all have the same value for VNCR_EL2 - 0, Generating an EL2 translation regime translation abort on use of the VNCR_EL2 register - 1, Reserved sign exteneded bits of VNCR_EL2 are same as bit[52] or bit[48] based on if large VA is supported or not, for all purposes other than reading back the register
unpred_zero_spsr_btype	bool	0x1	Constrained unpredictable control to make SPSR_ELx.BTYPE 0 instead of PSTATE.BTYPE on synchronous exceptions other than Software Step, PC alignment fault, Instruction Abort, Breakpoint or Address Matching Vector Catch, Illegal Execution State, BRK instruction, Branch Target
unpredictable_exclusive_abort_memtype	int	0x0	Cause MMU abort if exclusive access is not supported in certain memory type (0=exclusives allowed in all memory types, 1=exclusives abort in Device memory types, 2=exclusives abort in any type other than WB inner cacheable)
unpredictable_hvc_behaviour	int	0x0	HVC unpredictable behaviour. 0, UNDEF. 1, NOP.
unpredictable_smc_behaviour	int	0x0	SMC unpredictable behaviour. 0, UNDEF. 1, NOP.
use_Xt_as_LDG_STG_input	bool	0x1	Use new MTE Intructions formats for LDG/ST(Z)(2)G, which use Xt as an input
use_mte_eac_02_instructions_encoding	bool	0x1	Use new MTE Intructions encoding since MTE spec EAC 0.2 in ARMv8.5.
use_mte_eac_08_tfsr_encoding	bool	0x1	Use new MTE TFSR_ELx encodings since MTE spec EAC 0.8 in ARMv8.5.
use_rosetta_disass	int	0x1	Use Rosetta disassembly library. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
use_tlb_contig_hint	bool	0x0	Translation table entries with the contiguous hint bit set generate large TLB entries.
walk_cache_latency	int	0x0	Walk cache latency for TA (Timing Annotation), expressed in simulation ticks
warn_unpredictable_in_v7	bool	0x1	If true, behaviour which is unpredictable in V7 yet is predictable in V8 will produce a warning
watchpoint-log2secondary_restriction	int	0x0	log2 size of secondary restriction of FAR/EDWAR possible values on watchpoint hit for load/store operations.
wnr_is_read_for_s2f_on_s1_atomic_instr_fault	bool	0x0	Whether WnR is 0 for stage2 fault on stage1 for atomic instructions
wnr_is_read_for_s2f_on_s1_dbm_update	bool	0x0	Whether WnR is 0 for stage2 fault on stage1 descriptor dbm update

Related references

5.14 ScalableVectorExtension on page 5-2931

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

3.4.2 ARMAEMv8AMPCT

ARM AEM v8-A(MP) CPU component - number of core configurable at runtime. This model is written in C++.

ARMAEMv8AMPCT contains the following CADI targets:

- ARMAEMv8-A_MP
- Cluster_ARMAEMv8-A_MP
- PVCache
- TlbCadi

ARMAEMv8AMPCT contains the following MTI components:

- *4.4 ARM_AEMv8-A_MP - trace* on page 4-908
- *4.48 ARMv8Cluster - trace* on page 4-2401
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402
- *4.68 DSU - trace* on page 4-2463
- *4.73 GICv3CPUInterfaceDecoder - trace* on page 4-2489
- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.108 PVCache - trace* on page 4-2761
- *4.114 TLB - trace* on page 4-2834

About ARMAEMv8AMPCT

- AEMv8-A is deprecated and will be replaced by AEMvA, which enables both the existing Armv8-A architecture and Future Architecture Technologies. Support for architectural features is selectable through parameters. For additional information, contact *support@arm.com*.
- This component supports multiple versions of the Armv8-A architecture and provides parameters to enable or disable particular architectural features. Some of these parameters allow you to enable features in versions of the architecture earlier than the one in which they were introduced. These parameters have the possible values:

  0 Feature is not enabled.

  1 Feature is enabled, but only if the model is set to implement a version of the architecture in which the feature is supported.

  2 Feature is enabled, regardless of which architecture version the model implements.
- The number of cores is configured using the `NUM_CORES` parameter. The per-core parameters are preceded by `cpun`, where *n* identifies the core (0-7).

AEMv8A core personalities

The AEM core personalities feature allows you to configure AEM instances in a platform to use the IMPLEMENTATION DEFINED registers and UNPREDICTABLE behavior for a specific implementation of your choice at model startup.

Configuring an AEM with a core personality requires a license for both the AEM and the selected implementation.

Set the personality using the environment variable `FASTSIM_AEMV8_PROFILE`, or the parameter `impdef_regs_and_unpred_from_implementation`.

The parameter allows you to configure different instances in the same platform with different personalities, including subclusters in a heterogeneous AEM. The environment variable takes precedence over the parameter and affects all instances of the AEM in the platform. Otherwise the two options function identically.

To see the available values for the environment variable or parameter, set either of them to the special value `list`. The model prints the list of available values and exits. An example value is `ARM_Cortex-A57`. Then set the parameter or environment variable to the required value.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Configuring a core personality only affects IMPLEMENTATION DEFINED registers and UNPREDICTABLE behavior. In other respects, the cluster or subcluster still behaves like the AEM. For example, all parameters default to the AEM default values. Therefore, parameters have to be manually configured to valid values for the configured personality, as many of the defaults are incompatible with any given implementation.

To assist with this configuration, the AEM prints warnings for any parameter with an invalid value for the configured personality. The warning lists the parameter name and the valid value or range of values it can be set to for the selected personality.

Running the model with invalid parameter configurations for the selected personality can lead to unexpected behavior.

Mapping architectural features to Fast Models releases

The following table shows which Fast Models releases the key architectural features are implemented in, from Armv8.3-A onwards. Support is at EAC quality unless otherwise stated.

Table 3-74 Architectural features and Fast Models releases

Architecture version	Feature	Fast Models version									
		11.13	11.12	11.11	11.10	11.9	11.8	11.7	11.6	11.5	11.4
Armv8.3-A	All [h]	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Armv8.4-A	All [i]	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Armv8.5-A	Memory Tagging Extension (MTE) functionality	✓	✓	✓	✓	✓	✓	✓	✓ (Beta)	-	-
	MTE performance	✓	✓	✓	✓	✓	✓ (Beta)	-	-	-	-
	All except MTE [j].	✓	✓	✓	✓	✓	✓	✓	✓	✓	-
Armv8.6-A	All [k]	✓	✓	✓	✓	✓	-	-	-	-	-
Armv8.7-A	Limited TLBI maintenance	✓ (Alpha)	-	-	-	-	-	-	-	-	-
	All except TLBI maintenance [l]	✓	-	-	-	-	-	-	-	-	-
Future Architecture Technologies	SVE2, TME, CSRE, BRBE. [m]	✓	✓	✓	✓	✓	✓	-	-	-	-

[h] Armv8.3-A includes pointer authentication, nested virtualization, Advanced SIMD complex number support, improved Javascript data type conversion support, memory consistency model changes, support for larger system-visible caches, debug over powerdown.

[i] Armv8.4-A includes Secure EL2, additional cryptographic hashes, activity monitors, enhanced support for nested virtualization, Memory Partitioning and Monitoring (MPAM), stage 2 forced write-back, dot product instructions, data independent timing instructions, Large System Extensions (LSE), support for TLB maintenance instructions and for TLB range instructions, Translation Table Level (TTL), enhancements to weaker release consistency, debug relaxations and extensions, flag manipulation instructions.

[j] Armv8.5-A includes Branch Target Indicators (BTI), Enhanced Virtualization Traps (EVT), Random Number Generator (RNG), Flag Manipulation instructions, Floating-point round to integer, DC CVADP.

[k] Armv8.6-A includes General Matrix Multiply (GEMM), EnhancedPAC2, FPAC, Data Gathering Hint, Additional PMU events, Enhanced Counter Virtualization.

[l] Armv8.7-A includes WFE and WFI with timeouts, Larger physical address for 4KB and 16KB translation granules, MTE Asymmetric Fault Handling, Enhancements to Privilege Access Never (PAN), Enhancements to PMU and SPE.

[m] Implemented by AEMvA only.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-75 Ports

Name	Protocol	Type	Description
CNTHPIRQ[8]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTHPSIRQ[8]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTHVIRQ[8]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTHVSIRQ[8]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTPNSIRQ[8]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTPSIRQ[8]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTVIRQ[8]	*Signal* on page 2-85	Master	Timer signals to SOC.
CRITICALIRQ[8]	*Signal* on page 2-85	Master	RAS Critical Error Interrupt.
ERRORIRQ[8]	*Signal* on page 2-85	Master	RAS Error Recovery Interrupt.
FAULTIRQ[8]	*Signal* on page 2-85	Master	RAS Fault Handling Interrupt
acp_s	PVBus	Slave	AXI ACP slave port.
broadcastcachemaint	*Signal* on page 2-85	Slave	ACE defined pins.
broadcastinner	*Signal* on page 2-85	Slave	ACE defined pins.
broadcastouter	*Signal* on page 2-85	Slave	ACE defined pins.
cfgend[8]	*Signal* on page 2-85	Slave	This signal if for EE bit initialisation.
cfgsdisable	*Signal* on page 2-85	Slave	This signal disables write access to some secure Interrupt Controller registers.
clk_in	ClockSignal	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the core executes instructions.
clrexmonack	*Signal* on page 2-85	Master	Acknowledge handshake signal for the clrexmonreq signal
clrexmonreq	*Signal* on page 2-85	Slave	Signals the clearing of an external global exclusive monitor
clusterid	*Value* on page 2-85	Slave	The port reads the value in CPU ID register field, bits[11:8] of the MPIDR.
cntvalueb	*CounterInterface* on page 2-84	Slave	Interface to SoC level counter module.
commirq[8]	*Signal* on page 2-85	Master	Interrupt signal from debug communications channel.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-75 Ports (continued)

Name	Protocol	Type	Description
config64[8]	*Signal* on page 2-85	Slave	Register width after reset.
cp15sdisable[8]	*Signal* on page 2-85	Slave	This signal disables write access to some system control processor registers.
cpuporeset[8]	*Signal* on page 2-85	Slave	CPU power on reset. Initializes all the processor logic, including debug logic.
cryptodisable[8]	*Signal* on page 2-85	Slave	Disable cryptography extensions after reset.
cti[8]	*v8EmbeddedCrossTrigger_controlprotocol* on page 2-84	Master	Cross trigger matrix port.
cti0extin[4]	*Signal* on page 2-85	Slave	CTI trace inputs for core 0.
cti0extout[4]	*Signal* on page 2-85	Master	CTI trace outputs for core 0.
cti1extin[4]	*Signal* on page 2-85	Slave	CTI trace inputs for core 1.
cti1extout[4]	*Signal* on page 2-85	Master	CTI trace outputs for core 1.
cti2extin[4]	*Signal* on page 2-85	Slave	CTI trace inputs for core 2.
cti2extout[4]	*Signal* on page 2-85	Master	CTI trace outputs for core 2.
cti3extin[4]	*Signal* on page 2-85	Slave	CTI trace inputs for core 3.
cti3extout[4]	*Signal* on page 2-85	Master	CTI trace outputs for core 3.
cti4extin[4]	*Signal* on page 2-85	Slave	CTI trace inputs for core 4.
cti4extout[4]	*Signal* on page 2-85	Master	CTI trace outputs for core 4.
cti5extin[4]	*Signal* on page 2-85	Slave	CTI trace inputs for core 5.
cti5extout[4]	*Signal* on page 2-85	Master	CTI trace outputs for core 5.
cti6extin[4]	*Signal* on page 2-85	Slave	CTI trace inputs for core 6.
cti6extout[4]	*Signal* on page 2-85	Master	CTI trace outputs for core 6.
cti7extin[4]	*Signal* on page 2-85	Slave	CTI trace inputs for core 7.
cti7extout[4]	*Signal* on page 2-85	Master	CTI trace outputs for core 7.
ctidbgirq[8]	*Signal* on page 2-85	Master	Cross Trigger Interface (CTI) interrupt trigger output.
dbgen[8]	*Signal* on page 2-85	Slave	External debug interface.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-75 Ports (continued)

Name	Protocol	Type	Description
dbgnopwrdwn[8]	*Signal* on page 2-85	Master	Debug no power down request.
dbgpwrdwnack[8]	*Signal* on page 2-85	Master	Debug power down acknowledge.
dbgpwrdwnreq[8]	*Signal* on page 2-85	Slave	Debug power down request.
dbgpwrupreq[8]	*Signal* on page 2-85	Master	Debug power up request.
dev_debug_s	PVBus	Slave	External debug interface.
etm[8]	PVBus	Master	Embedded trace macrocell port.
event	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE.
external_trace_reset[8]	*Signal* on page 2-85	Slave	ETMv4 External Trace Reset signal.
fiq[8]	*Signal* on page 2-85	Slave	This signal drives the CPUs fast-interrupt handling.
gicv3_redistributor_s[8]	*GICv3Comms* on page 2-84	Slave	GICv3 cpu interface ports.
irq[8]	*Signal* on page 2-85	Slave	This signal drives the CPUs interrupt handling.
irqs[224]	*Signal* on page 2-85	Slave	These signals drive the CPU's interrupt controller interrupt lines.
l2reset	*Signal* on page 2-85	Slave	This signal resets timer and interrupt controller.
memorymapped_debug_s	PVBus	Slave	External debug interface.
niden[8]	*Signal* on page 2-85	Slave	External debug interface.
periphbase	*Value_64* on page 2-85	Slave	This port sets the base address of private peripheral region.
pmbirq[8]	*Signal* on page 2-85	Master	Interrupt signal from the statistical profiling unit.
pmuirq[8]	*Signal* on page 2-85	Master	Interrupt signal from performance monitoring unit.
presetdbg[8]	*Signal* on page 2-85	Slave	Initialize the shared debug APB, Cross Trigger Interface (CTI), and Cross Trigger Matrix (CTM) logic.
pvbus_m0	PVBus	Master	The core will generate bus requests on this port.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-75 Ports (continued)

Name	Protocol	Type	Description
rei[8]	*Signal* on page 2-85	Slave	Individual processor RAM Error Interrupt signal input.
reset[8]	*Signal* on page 2-85	Slave	Raising this signal will put the core into reset mode.
romaddr	*Value_64* on page 2-85	Slave	Debug ROM base address.
romaddrv	*Signal* on page 2-85	Slave	Debug ROM base address valid.
rvbar[8]	*Value_64* on page 2-85	Slave	Reset vector base address.
sei[8]	*Signal* on page 2-85	Slave	Per core System Error physical pins.
smpnamp[8]	*Signal* on page 2-85	Master	This signals AMP or SMP mode for each core.
spiden[8]	*Signal* on page 2-85	Slave	External debug interface.
spniden[8]	*Signal* on page 2-85	Slave	External debug interface.
standbywfe[8]	*Signal* on page 2-85	Master	This signal indicates if a core is in WFE state.
standbywfi[8]	*Signal* on page 2-85	Master	This signal indicates if a core is in WFI state.
standbywfil2	*Signal* on page 2-85	Master	This signal indicated all cores and L2 are in a power down state
teinit[8]	*Signal* on page 2-85	Slave	This signal provides default exception handling state.
ticks[8]	*InstructionCount* on page 2-84	Master	This port should be connected to one of the two ticks ports on a 'visualisation' component, in order to display a running instruction count.
trace_unit_reset[8]	*Signal* on page 2-85	Slave	ETMv4 Trace Unit Reset signal.
vcpumntirq[8]	*Signal* on page 2-85	Master	Interrupt signal for virtual CPU maintenance IRQ.
vfiq[8]	*Signal* on page 2-85	Slave	Virtual FIQ input. Note that the irq/fiq pins are wired directly to the core if there is no internal VGIC. If there is an internal VGIC then these are ignored.
vinithi[8]	*Signal* on page 2-85	Slave	This signal controls of the location of the exception vectors at reset.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-75 Ports (continued)

Name	Protocol	Type	Description
virq[8]	*Signal* on page 2-85	Slave	Virtual IRQ input. Note that the irq/fiq pins are wired directly to the core if there is no internal VGIC. If there is an internal VGIC then these are ignored.
virtio_s	PVBus	Slave	The virtio coherent port, hooks directly into the L2 system and becomes coherent (assuming attributes are set correctly).
vsei[8]	*Signal* on page 2-85	Slave	Processor Virtual System Error Interrupt request.

Table 3-76 Parameters for Cluster_ARMAEMv8-A_MP

Name	Type	Default value	Description
AIDR	int	0x0	Value of AIDR_EL1 register.
AMPIDR	int	0x400bbb000	Value of AMU Peripheral Identification Register
BPIMVA_causes_translation_lookup	bool	0x0	Do a translation when BPIMVA instruction is executed (which may cause a translation fault).
CCSIDR-L1D_override	int	0x0	If nonzero, override the value presented in CCSIDR for L1D (this is cosmetic and does not affect cache behaviour).
CCSIDR-L1I_override	int	0x0	If nonzero, override the value presented in CCSIDR for L1I (this is cosmetic and does not affect cache behaviour).
CCSIDR-L2_override	int	0x0	If nonzero, override the value presented in CCSIDR for L2 (this is cosmetic and does not affect cache behaviour).
CCSIDR-L3_override	int	0x0	If nonzero, allow L3 selection in CSSELR and present this value in CCSIDR (this is cosmetic and does not affect cache behaviour).
CPUCFR	int	0x0	Value of CPU Configuration Register
CTIPIDR	int	0x0	If non-zero, override the CTI Peripheral Identification Register
DBGPIDR	int	0x0	If non-zero, override the Debug Peripheral Identification Register
ERXMISC0_mask	int	0x0	Write Mask for ERXMISC0 RAS Register
GMID-log2-block-size	int	0x4	Log2 of the block size accessed by STGM/LDGM/STZGM instructions
ISV_set_to_0_for_stage2_synch_external_abort	bool	0x0	Whether ESR_EL2.ISV is set to 0 on stage 2 synchronous external aborts
MIDR	int	0x410fd0f0	Value of MIDR_EL1 register.
NUM_CORES	int	0x1	Number of cores per cluster

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-76 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
PA_SIZE	int	0x28	Physical address range supported. For ARMv8.0 and ARMv8.1 this is limited to 48 bits.
PMUPIDR	int	0x0	If non-zero, override the PMU Peripheral Identification Register
abort_execution_from_device_memory	bool	0x0	Execution from device memory generates a prefetch abort.
advsimd_bf16_support_level	int	0x0	Implement BFloat16 operations from ARMv8.6. AArch64 Advanced SIMD and FP BFloat16 instructions are automatically enabled when has_arm_v8-6 is true. - 0, Not implemented. - 1, AArch64 Advanced SIMD and FP BFloat16 instructions only. - 2, AArch32 Advanced SIMD and VFP BFloat16 instructions only. - 3, Both AArch64 Advanced SIMD and FP and AArch32 Advanced SIMD and VFP BFloat16 instructions.
advsimd_i8mm_support_level	int	0x0	Implement Int8 matrix multiply operations from ARMv8.6. AArch64 Advanced SIMD and FP Int8 matrix multiply instructions are automatically enabled when has_arm_v8-6 is true. - 0, Not implemented. - 1, AArch64 Advanced SIMD and FP Int8 matrix multiply instructions only. - 2, AArch32 Advanced SIMD and VFP Int8 matrix multiply instructions only. - 3, Both AArch64 Advanced SIMD and FP and AArch32 Advanced SIMD and VFP Int8 matrix multiply instructions.
amair_reg_rw_mask	int	0x0	RW mask for implementation-defined registers.
amu_aux_voffset_mask	int	0x0	If ARMv8.6 is implemented, each bit of the field, 0 to 15, when 1 indicates that the corresponding virtual offset register, AMEVCNTVOFF1<n>_EL2, is implemented
amu_has_external_interface	int	0x0	Implement external memory-mapped access to system register of activity monitor unit from ARMv8.4. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
amu_has_sysreg_interface	bool	0x1	Implement system register access to activity monitor unit from ARMv8.4.
amu_reset_domain	int	0x0	Reset domain for activity monitor unit. 0, COLD_RESET. 1, WARM_RESET. 2, NONE.
amu_version	int	0x1	Selects the activity monitor version implemented - 1, AMUv1 for Armv8.4 is implemented. - 2, AMUv1 for Armv8.6 is implemented.
apsr_read_restrict	bool	0x0	At EL0, unknown bits of APSR are RAZ.
arm_v8_7_accelerator_support_level	int	0x0	Implements Armv8.7 accelerator support instructions LD/ST64B, ST64BV, ST64BV0, etc.
auxilliary_feature_register0	int	0x0	Value of AFR0 ID register.
cache-log2linelen	int	0x6	Log2 of the cache line length in bytes.
cache_maintenance_hits_watchpoints	bool	0x0	DCIMVA operations executed in AArch32 modes hit watchpoints.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-76 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
changing_block_size_without_bbm_support	int	0x0	Level of support for changing block size without break-before-make.
clear_IT_when_IL_set	bool	0x0	Clear IT bits when performing a *legal* exception return to AArch32 when IL is set
clear_IT_when_IL_set_explicitly	bool	0x0	Apart from clear_IT_when_IL_set, also clear IT bits when loading CPSR from SPSR/memory and IL == 1 in the value being loaded.
clear_reg_top_eret	int	0x1	Behaviour of the upper 32-bits of the Xn registers when changing between AArch32 state and AArch64 state. 0, upper 32-bits preserved for all registers. 1, upper 32-bits set to 0 for all accessible registers. 2, upper 32-bits set to 0 for a random selection of accessible registers. 3, upper-32-bits set to 0 for registers touched in AArch32
cpi_div	int	0x1	Divider for calculating CPI (Cycles Per Instruction)
cpi_mul	int	0x1	Multiplier for calculating CPI (Cycles Per Instruction)
dc_fault_unaligned_s1_device_s2_fwb	bool	0x0	Whether takes an Alignment Fault caused by the memory type on a DC {ZVA,GZVA,GVA} if the stage 1 memory type is any Device memory type
dcache-hit_latency	int	0x0	L1 D-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
dcache-maintenance_latency	int	0x0	L1 D-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
dcache-miss_latency	int	0x0	L1 D-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
dcache-prefetch_enabled	bool	0x0	Enable simulation of data cache prefetching. This is only used when dcache-state_modelled=true
dcache-read_access_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per access (of size dcache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
dcache-read_bus_width_in_bytes	int	0x8	L1 D-Cache read bus width in bytes used to calculate per-access timing annotations
dcache-read_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per byte accessed.dcache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-76 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
dcache-size	int	0x8000	L1 D-Cache size in bytes.
dcache-snoop_data_transfer_latency	int	0x0	L1 D-Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
dcache-ways	int	0x2	L1 D-Cache number of ways (sets are implicit from size).
dcache-write_access_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per access (of size dcache-write_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-write_latency is set. This is only used when dcache-state_modelled=true.
dcache-write_bus_width_in_bytes	int	0x8	L1 D-Cache write bus width in bytes used to calculate per-access timing annotations
dcache-write_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per byte accessed. dcache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
dcimva_requires_s2_write_permissions	bool	0x0	Data-cache invalidate by MVA operations require stage 2 write permission (virtualised AArch32 guest).
dczva_reports_lowest_addr_on_tag_check_fail	bool	0x0	Whether DC ZVA reports lowest address in FAR on tag check fail
delayed_dbgreg_between_secure_views	int	0x1	If delayed_dbgreg is enabled, whether the secure and nonsecure external views require explicit synchronization. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
dic-spi_count	int	0x40	Number of shared peripheral interrupts implemented
disable_sve_plugin	bool	0x0	If true, SVE will not be implemented in this processor even if the plugin is loaded
disable_unknown_update_event_on_reset	bool	0x0	Disables SYSREG_UPDATE event notification on reset for the registers whose bitfields are all reserverd or resets to architecturally unknown value
ecv_support_level	int	0x0	Implement Enhanced Counter Virtualization feature from ARMv8.6. 0, Not supported. 1, fully supported without CNTPOFF. 2, fully supported with CNTPOFF.
el0_can_access_imp_def_functionality	bool	0x0	If not made UNDEF by imp_def_functionality_behaviour, EL0 can access IMPLEMENTATION DEFINED registers and system instructions.
el0_el1_only_non_secure	bool	0x0	Secure/non-secure state if EL2 and EL3 are not implemented. 0, secure. 1, non-secure.
enable_tlb_contig_check	bool	0x0	Perform extra pagetable walks to check translation table entries that have the contiguous hint bit set.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-76 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
enhanced_pac2_level	int	0x0	Implements Enhanced PAC2 from ARMv8.6. This feature is mandatory for ARMv8.6 but can be cherrypicked to a ARMv8.3(or greater) implementation. 0: No EnhancedPAC2, 1: EnhancedPAC2 Only, 2: EnhancedPAC2 with FPAC, 3: EnhancedPAC2 with FPACCombined.
error_record_feature_register	string	""	RAS feature register values. An array of JSON objects. The JSON schema for the array is: [{"ED":0x0,"IMPDEF_3_2":0x0,"UI":0x0,"FI":0x0,"UE":0x0,"CFI":0x0,"CEC":0x0,"RP":0x0,"DUI":0x0,"CEO":0x0,"CI":0x0,"TS":0x0,"INJ":0x0,"FRX":0x0,"UC":0x0,"UEU":0x0,"UER":0x0,"UEO":0x0,"DE":0x0,"CE":0x0,"Visibility":"Core"},other_feature_register_values]. Where ED,UI,FI,CE and UE have valid values betwn 0x0 - 0x3. CFI and DUI have valid values 0x0, 0x2 and 0x3. CEC has valid values 0x0,0x2 or 0x4. RP,CEO,INJ,FRX,UC,UEU,UER,UEO,DE has valid values 0x0 or 0x1. CI and TS has valid values of 0x0, 0x1 and 0x2. Visibility has valid values "Core" or "Cluster"
error_record_feature_register_json_file	string	""	File path to the RAS feature register values as JSON. The file uses the same format as the error_record_feature_register parameter value
exercise_stxr_fail	bool	0x0	Reject a pseudo-random majority of exclusive store instructions
ext_abort_device_GRE_prefetch_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_index, Valid indices in range [0, number_of_error_records).
ext_abort_device_GRE_prefetch_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_device_GRE_read_is_critical	bool	0x0	Critical reporting of device-GRE read external aborts.
ext_abort_device_GRE_read_is_sync	int	0x2	Synchronous reporting of device-GRE read external aborts. 0, asynchronous. 1, synchronous. 2, same as ext_abort_device_read_is_sync.
ext_abort_device_GRE_read_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_device_read_ras_index, Valid indices in range [0, number_of_error_records).
ext_abort_device_GRE_read_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_device_read_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_device_GRE_write_is_critical	bool	0x0	Critical reporting of device-GRE write external aborts
ext_abort_device_GRE_write_is_sync	int	0x2	Synchronous reporting of device-GRE write external aborts. 0, asynchronous. 1, synchronous. 2, same as ext_abort_device_write_is_sync.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-76 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
ext_abort_device_GRE_write_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_device_write_ras_index, Valid indices in range [0, number_of_error_records).
ext_abort_device_GRE_write_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_device_write_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_device_nGRE_prefetch_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_index, Valid indices in range [0, number_of_error_records).
ext_abort_device_nGRE_prefetch_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_device_nGRE_read_is_critical	bool	0x0	Critical reporting of device-nGRE read external aborts.
ext_abort_device_nGRE_read_is_sync	int	0x2	Synchronous reporting of device-nGRE read external aborts. 0, asynchronous. 1, synchronous. 2, same as ext_abort_device_read_is_sync.
ext_abort_device_nGRE_read_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_device_read_ras_index, Valid indices in range [0, number_of_error_records).
ext_abort_device_nGRE_read_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_device_read_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_device_nGRE_write_is_critical	bool	0x0	Critical reporting of device-nGRE write external aborts
ext_abort_device_nGRE_write_is_sync	int	0x2	Synchronous reporting of device-nGRE write external aborts. 0, asynchronous. 1, synchronous. 2, same as ext_abort_device_write_is_sync.
ext_abort_device_nGRE_write_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_device_write_ras_index, Valid indices in range [0, number_of_error_records).
ext_abort_device_nGRE_write_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_device_write_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_device_prefetch_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_index, Valid indices in range [0, number_of_error_records).
ext_abort_device_prefetch_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_device_read_is_critical	bool	0x0	Critical reporting of device-nGnRE read external aborts
ext_abort_device_read_is_sync	bool	0x1	Synchronous reporting of device-nGnRE read external aborts

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-76 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
ext_abort_device_read_ras_index	int	0x0	External Aborts are reported in RAS record index specified in this param. Values: Valid indices in range [0, number_of_error_records).
ext_abort_device_read_ras_type	int	0x0	External Aborts are reported as RAS error type specified in this param. Values: 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_device_write_is_critical	bool	0x0	Critical reporting of device-nGnRE write external aborts
ext_abort_device_write_is_sync	bool	0x0	Synchronous reporting of device-nGnRE write external aborts
ext_abort_device_write_ras_index	int	0x0	External Aborts are reported in RAS record index specified in this param. Values: Valid indices in range [0, number_of_error_records).
ext_abort_device_write_ras_type	int	0x0	External Aborts are reported as RAS error type specified in this param. Values: 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_fill_data	int	-0x202020303020203	Returned data, if external aborts are asynchronous
ext_abort_normal_cacheable_read_is_critical	bool	0x0	Critical reporting of normal write-back cacheable-read external aborts
ext_abort_normal_cacheable_read_is_sync	bool	0x1	Synchronous reporting of normal write-back cacheable-read external aborts
ext_abort_normal_cacheable_read_ras_index	int	0x0	External Aborts are reported in RAS record index specified in this param. Values: Valid indices in range [0, number_of_error_records).
ext_abort_normal_cacheable_read_ras_type	int	0x0	External Aborts are reported as RAS error type specified in this param. Values: 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_normal_cacheable_write_is_critical	bool	0x0	Critical reporting of normal write-back cacheable write external aborts
ext_abort_normal_cacheable_write_is_sync	bool	0x0	Synchronous reporting of normal write-back cacheable write external aborts
ext_abort_normal_cacheable_write_ras_index	int	0x0	External Aborts are reported in RAS record index specified in this param. Values: Valid indices in range [0, number_of_error_records).
ext_abort_normal_cacheable_write_ras_type	int	0x0	External Aborts are reported as RAS error type specified in this param. Values: 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_normal_noncacheable_prefetch_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_index, Valid indices in range [0, number_of_error_records).

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-76 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
ext_abort_normal_noncacheable_prefetch_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_normal_noncacheable_read_is_critical	bool	0x0	Critical reporting of normal noncacheable-read external aborts
ext_abort_normal_noncacheable_read_is_sync	bool	0x1	Synchronous reporting of normal noncacheable-read external aborts
ext_abort_normal_noncacheable_read_ras_index	int	0x0	External Aborts are reported in RAS record index specified in this param. Values: Valid indices in range [0, number_of_error_records).
ext_abort_normal_noncacheable_read_ras_type	int	0x0	External Aborts are reported as RAS error type specified in this param. Values: 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_normal_noncacheable_write_is_critical	bool	0x0	Critical reporting of normal noncacheable write external aborts
ext_abort_normal_noncacheable_write_is_sync	bool	0x0	Synchronous reporting of normal noncacheable write external aborts
ext_abort_normal_noncacheable_write_ras_index	int	0x0	External Aborts are reported in RAS record index specified in this param. Values: Valid indices in range [0, number_of_error_records).
ext_abort_normal_noncacheable_write_ras_type	int	0x0	External Aborts are reported as RAS error type specified in this param. Values: 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_normal_wt_cacheable_prefetch_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_index, Valid indices in range [0, number_of_error_records).
ext_abort_normal_wt_cacheable_prefetch_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_normal_wt_cacheable_read_is_critical	bool	0x0	Critical reporting of normal write-through cacheable-read external aborts
ext_abort_normal_wt_cacheable_read_is_sync	int	0x2	Synchronous reporting of normal write-through read external aborts. 0, asynchronous. 1, synchronous. 2, same as ext_abort_normal_cacheable_read_is_sync.
ext_abort_normal_wt_cacheable_read_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_normal_cacheable_read_ras_index, Valid indices in range [0, number_of_error_records).
ext_abort_normal_wt_cacheable_read_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_normal_cacheable_read_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-76 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
ext_abort_normal_wt_cacheable_write_is_critical	bool	0x0	Critical reporting of normal write-through write external aborts
ext_abort_normal_wt_cacheable_write_is_sync	int	0x2	Synchronous reporting of normal write-through write external aborts. 0, asynchronous. 1, synchronous. 2, same as ext_abort_normal_cacheable_write_is_sync.
ext_abort_normal_wt_cacheable_write_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_normal_cacheable_write_ras_index, Valid indices in range [0, number_of_error_records).
ext_abort_normal_wt_cacheable_write_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_normal_cacheable_write_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_prefetch_device_GRE_read_is_critical	bool	0x0	Critical reporting of external aborts generated by device-GRE instruction fetches
ext_abort_prefetch_device_GRE_read_is_sync	int	0x2	Behaviour of external aborts generated by device-GRE instruction fetches. 0, asynchronous abort. 1, synchronous abort. 2, same as ext_abort_prefetch_is_sync.
ext_abort_prefetch_device_nGRE_read_is_critical	bool	0x0	Critical reporting of external aborts generated by device-nGRE instruction fetches
ext_abort_prefetch_device_nGRE_read_is_sync	int	0x2	Behaviour of external aborts generated by device-nGRE instruction fetches. 0, asynchronous abort. 1, synchronous abort. 2, same as ext_abort_prefetch_is_sync.
ext_abort_prefetch_device_read_is_critical	bool	0x0	Critical reporting of external aborts generated by device-nGnRE instruction fetches
ext_abort_prefetch_device_read_is_sync	int	0x2	Behaviour of external aborts generated by device-nGnRE instruction fetches. 0, asynchronous abort. 1, synchronous abort. 2, same as ext_abort_prefetch_is_sync.
ext_abort_prefetch_is_critical	bool	0x0	Critical reporting of external aborts generated by normal writeback cacheable instruction fetches
ext_abort_prefetch_is_sync	bool	0x1	Behaviour of external aborts generated by normal writeback cacheable instruction fetches. 0, asynchronous abort. 1, synchronous abort.
ext_abort_prefetch_noncacheable_read_is_critical	bool	0x0	Critical reporting of external aborts generated by normal noncacheable instruction fetches
ext_abort_prefetch_noncacheable_read_is_sync	int	0x2	Behaviour of external aborts generated by normal noncacheable instruction fetches. 0, asynchronous abort. 1, synchronous abort. 2, same as ext_abort_prefetch_is_sync.
ext_abort_prefetch_ras_index	int	0x0	External Aborts are reported in RAS record index specified in this param. Values: Valid indices in range [0, number_of_error_records).

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-76 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
ext_abort_prefetch_ras_type	int	0x0	External Aborts are reported as RAS error type specified in this param. Values: 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_prefetch_so_read_is_critical	bool	0x0	Critical reporting of external aborts generated by device-nGnRnE instruction fetches
ext_abort_prefetch_so_read_is_sync	int	0x2	Behaviour of external aborts generated by device=nGnRnE instruction fetches. 0, asynchronous abort. 1, synchronous abort. 2, same as ext_abort_prefetch_is_sync.
ext_abort_prefetch_wt_cacheable_read_is_critical	bool	0x0	Critical reporting of external aborts generated by normal writethrough cacheable instruction fetches
ext_abort_prefetch_wt_cacheable_read_is_sync	int	0x2	Behaviour of external aborts generated by normal writethrough cacheable instruction fetches. 0, asynchronous abort. 1, synchronous abort. 2, same as ext_abort_prefetch_is_sync.
ext_abort_so_prefetch_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_index, Valid indices in range [0, number_of_error_records).
ext_abort_so_prefetch_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_so_read_is_critical	bool	0x0	Critical reporting of device-nGnRnE read external aborts
ext_abort_so_read_is_sync	bool	0x1	Synchronous reporting of device-nGnRnE read external aborts
ext_abort_so_read_ras_index	int	0x0	External Aborts are reported in RAS record index specified in this param. Values: Valid indices in range [0, number_of_error_records).
ext_abort_so_read_ras_type	int	0x0	External Aborts are reported as RAS error type specified in this param. Values: 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_so_write_is_critical	bool	0x0	Critical reporting of device-nGnRnE write external aborts
ext_abort_so_write_is_sync	bool	0x1	Synchronous reporting of device-nGnRnE write external aborts
ext_abort_so_write_ras_index	int	0x0	External Aborts are reported in RAS record index specified in this param. Values: Valid indices in range [0, number_of_error_records).
ext_abort_so_write_ras_type	int	0x0	External Aborts are reported as RAS error type specified in this param. Values: 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_ttw_cacheable_read_is_critical	bool	0x0	Critical reporting of TTW cacheable read external aborts
ext_abort_ttw_cacheable_read_is_sync	bool	0x1	Synchronous reporting of TTW cacheable read external aborts
ext_abort_ttw_cacheable_read_ras_index	int	0x0	External Aborts are reported in RAS record index specified in this param. Values: Valid indices in range [0, number_of_error_records).

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-76 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
ext_abort_ttw_cacheable_read_ras_type	int	0x0	External Aborts are reported as RAS error type specified in this param. Values: 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_ttw_noncacheable_read_is_critical	bool	0x0	Critical reporting of TTW noncacheable read external aborts
ext_abort_ttw_noncacheable_read_is_sync	bool	0x1	Synchronous reporting of TTW noncacheable read external aborts
ext_abort_ttw_noncacheable_read_ras_index	int	0x0	External Aborts are reported in RAS record index specified in this param. Values: Valid indices in range [0, number_of_error_records).
ext_abort_ttw_noncacheable_read_ras_type	int	0x0	External Aborts are reported as RAS error type specified in this param. Values: 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_ttw_wt_cacheable_read_is_critical	bool	0x0	Critical reporting of TTW write-through cacheable read external aborts
ext_abort_ttw_wt_cacheable_read_is_sync	int	0x2	Synchronous reporting of TTW write-through cacheable read external aborts. 0, asynchronous. 1, synchronous. 2, same as ext_abort_ttw_cacheable_read_is_sync.
ext_abort_ttw_wt_cacheable_read_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_ttw_cacheable_read_ras_index, Valid indices in range [0, number_of_error_records).
ext_abort_ttw_wt_cacheable_read_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_ttw_cacheable_read_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
fault_on_nT_bit_set	bool	0x1	Whether block translation table entries with the nT bit set should always fault. Only applies when changing_block_size_without_bbm_support_level is 1 or higher.
fault_unalign_to_unsupported_mem	int	0x0	If has_unaligned_single_copy_atomicity is true, whether unaligned A64 atomic, exclusive and acquire/release instructions to non iWB-oWB generate fault. This parameter is a bitmap. Bits 0,1,2 should be set accordingly to enable the fault behaviour. bit 0(LSB): atomic access should fault, bit 1: exclusive access should fault, bit 2: acquire/release should fault.
fault_unaligned_s1_device_s2_fwb	bool	0x0	Whether unaligned access with stage1 Device memory and final memory attribute forced to normal by FWB can generate alignment fault
force_deterministic_irg_tag_generation	bool	0x0	Force the random tag generated by the IRG instruction when GCR_EL1.RRND=1 to equal RGSR_EL1.SEED[3:0] rather than a non-deterministic value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-76 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
fsr_ext_bit_update_kind	int	0x3	Set/Clear DFSR/IFSR EA bit on Synchronous/Async External Aborts. 0: Never Set, 1: Set on Synchronous Ext Aborts 2: Set on Asynchronous Ext Aborts 3: Set on both Sync and Async Ext Aborts.
global_debug_rom.ROMDEVID	int	0x0	Value of Debug Rom Device Identification Register
global_debug_rom.ROMPIDR	int	0x4000 bb000	Value of Debug Rom Peripheral Identification Register
hardware_translation_table_update_implemented	int	0x1	Implement hardware translation table updates from ARMv8.1. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.1 is enabled. - 2, feature is implemented.
has_16bit_asids	bool	0x1	Enable 16-bit ASIDs.
has_16bit_vmids	int	0x1	Implement support for 16-bit VMIDs from ARMv8.1. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.1 is enabled. - 2, feature is implemented.
has_16k_granule	bool	0x0	Implement the 16k LPAE translation granule.
has_4k_granule	bool	0x1	Implement the 4k LPAE translation granule.
has_52bit_address_with_16k	int	0x0	Implements Armv8.7 52-bit IPA/PA support for 16k Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if Armv8.7 is enabled. - 2, feature is implemented.
has_52bit_address_with_4k	int	0x0	Implements Armv8.7 52-bit IPA/PA support for 4k Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if Armv8.7 is enabled. - 2, feature is implemented.
has_64k_granule	bool	0x1	Implement the 64k LPAE translation granule.
has_aarch32_hpd	bool	0x0	If true then hierarchical permission disable is supported in AArch32
has_aarch64	bool	0x1	All implemented exception levels can run in AArch64
has_actlr2	bool	0x0	If true ACLTR2 exists and ACTLR2(NS) is aliased to ACTLR_EL1[63:32]
has_amu	int	0x0	Implement activity monitor functionality from ARMv8.4. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_arm_v8-1	bool	0x0	Implement the ARMv8.1 Extension.
has_arm_v8-2	bool	0x0	Implement the ARMv8.2 Extension.
has_arm_v8-3	bool	0x0	Implement the ARMv8.3 Extension.
has_arm_v8-4	bool	0x0	Implement the ARMv8.4 Extension.
has_arm_v8-5	bool	0x0	Implement the ARMv8.5 Extension.
has_arm_v8-6	bool	0x0	Implement the ARMv8.6 Extension.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-76 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
has_arm_v8-7	bool	0x0	Implement the Armv8.7 Extension. This feature is incomplete and under development.
has_at_with_pan	int	0x1	Implement new AT instructions with PAN support. Possible values of this parameter are: - 1, feature is implemented if ARMv8.2 is enabled. - 2, feature is implemented.
has_axflag_xaflag	int	0x1	Implement flag manipulation instructions (AXFlag, XAFlag) from ARMv8.5. Possible values of this parameter are: - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
has_axflag_xaflag_frint	int	0x1	Implement flag manipulation instructions (AXFlag, XAFlag) and floating-point rounding to int instructions (FRINT[32|64][X|Z]) from ARMv8.5. If this parameter is enabled, it also enables both has_axflag_xaflag and has_frint. If support for only one of the features is needed, please use the individual parameters and do not enable this one. Possible values of this parameter are: - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
has_branch_target_exception	int	0x0	Implement Branch target identification mechanism from ARMv8.5. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
has_ccidx	bool	0x0	Implement the ARMv8.3 CCSIDR Extension. Extending the ccsidr number of sets.
has_cfinv_rmif_setf	int	0x1	Implement flag manipulation (CFINV, RMIF, SETF8, SETF16) instructions from ARMv8.4. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_coherent_icache	bool	0x0	Whether icache invalidation to the point of unification is required for instruction to data coherence. true - Invalidate operations not required
has_common_not_private_translations	int	0x1	Implement the TTBRn_ELx.CnP (Common not Private) controls from ARMv8.2. Possible values of this parameter are: - 1, feature is implemented if ARMv8.2 is enabled. - 2, feature is implemented.
has_complex_number	int	0x1	Implement ARMv8.3 complex number support, Multiply Accumulate and Add instructions. Possible values of this parameter are: - 1, feature is implemented if ARMv8.3 is enabled. - 2, feature is implemented.
has_cvadp_support	int	0x0	Implement instruction to support cache clean by deep persistence (DC CVADP) from ARMv8.5, can be selected for core implemented on any arch version starting ARMv8.2 Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.2 is enabled. - 2, feature is implemented.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-76 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
has_data_alignment_flag	int	0x0	Implement non-optimal misalignment flag for PMU/SPE from ARMv8.5 Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
has_delayed_ctireg	bool	0x0	Delay the functional effect of CTI register writes until ISB or implicit barrier.
has_delayed_dbgreg	bool	0x0	Delay the functional effect of external debug register writes until ISB or implicit barrier.
has_delayed_sysreg	bool	0x0	Delay the functional effect of system register writes until ISB or implicit barrier.
has_delayed_wfe_trap	int	0x0	Implements Configurable Delayed WFE trapping from ARMv8.6 Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.6 is enabled. - 2, feature is implemented.
has_dgh	int	0x1	Implements Data Gathering Hint instruction from ARMv8.6 Possible values of this parameter are: - 1, feature is implemented if ARMv8.6 is enabled. - 2, feature is implemented.
has_dot_product	int	0x1	Implement the dot product (UDOT, SDOT) instructions from ARMv8.4. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_e0pd	int	0x1	Implement ARMv8.5 feature to prevent unprivileged access to one half of the memory Possible values of this parameter are: - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
has_el2	bool	0x1	Implements EL2
has_el3	bool	0x1	Implements EL3
has_enhanced_pac	bool	0x0	If pointer authentication is enabled then implement enhanced PAC.
has_enhanced_pan	int	0x1	Implements Armv8.7 Enhanced PAN feature Possible values of this parameter are: - 1, feature is implemented if Armv8.7 is enabled. - 2, feature is implemented.
has_ets	bool	0x0	Whether Enhanced Translation Synchronization is supported
has_extended_recp_estimate	int	0x0	Implements increased precision of reciprocal instructions.) Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if Armv8.7 is enabled. - 2, feature is implemented.
has_fgt	int	0x1	Implements Fine-grained Virtualization Traps extension from ARMv8.6 Possible values of this parameter are: - 1, feature is implemented if ARMv8.6 is enabled. - 2, feature is implemented.
has_fp16	int	0x1	Implement the half-precision floating-point data processing instructions from ARMv8.2. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.2 is enabled. - 2, feature is implemented.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-76 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
has_fp16_fmlal	int	0x1	Implement the New Floating Point Multiplication Variant (FP16 FMLAL, FMLSL) instructions from ARMv8.4. Only supported if has_fp16=0x1. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_frint	int	0x1	Implement floating-point rounding to int instructions (FRINT[32|64][X|Z]) from ARMv8.5. Possible values of this parameter are: - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
has_generic_authentication	int	0x1	Implement ARMv8.3 generic authentication. Possible values of this parameter are: - 1, feature is implemented if ARMv8.3 is enabled. - 2, feature is implemented.
has_guest_translation_granule	int	0x1	Implement mechanism for guest translation granule identification from ARMv8.5, ID values determined by stage1 granule configuration parameters Possible values of this parameter are: - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
has_hardware_translation_table_update	int	0x2	Type of hardware translation table supported (when enabled by hardware_translation_table_update_implemented). 0, not implemented. 1, access bit updates implemented. 2, access bit updates and dirty bit mechanism implemented.
has_hcrx_el2	int	0x1	Implements new HCRX_EL2 id register from Armv8.7 Possible values of this parameter are: - 1, feature is implemented if Armv8.7 is enabled. - 2, feature is implemented.
has_id_reg_read	int	0x1	Implement read access to the ID registers (ESR_ELx.EC=0x18) Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_iesb	int	0x1	Implement support for implicit error sync event from ARMv8.2. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.2 is enabled. - 2, feature is implemented.
has_jscvt	int	0x1	Implement ARMv8.3 javascript Floating-point to Integer conversion instruction. Possible values of this parameter are: - 1, feature is implemented if ARMv8.3 is enabled. - 2, feature is implemented.
has_large_system_ext	bool	0x0	Implement the ARMv8 Large System Extensions.
has_large_va	int	0x0	Implement support for the extended 52-bit virtual addresses from ARMv8.2. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.2 is enabled. - 2, feature is implemented.
has_ldapur_stlur	int	0x1	Implement support for LDAPR and STLR instructions with immediate offsets from ARMv8.4. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-76 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
has_ldm_stm_ordering_control	int	0x0	Implement the SCTLR_ELx.LSMAOE (Load/Store Multiple Atomicity and Ordering Enable) and SCTLR_ELx,nTLSMD (no Trap Load/Store Multiple to Device) controls from ARMv8.2. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.2 is enabled. - 2, feature is implemented.
has_lrcpc	bool	0x0	If true then it support the RCpc feature (ARMv8.3)
has_mpam	int	0x0	Implement ARMv8.4 MPAM Registers and associated functionality. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_mt_pmu_disable_feature	int	0x0	Implements Multi-threading PMU disable extension from ARMv8.6 Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.6 is enabled. - 2, feature is implemented.
has_mte_ctrl_bits_stateful	bool	0x0	if memory_tagging_support_level == 1, Whether mte specific control bits in system registers are stateful
has_nested_virtualization	int	0x1	Implement ARMv8.3 nested virtualization. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.3 is enabled. - 2, feature is implemented.
has_no_os_double_lock	int	0x0	Do not implement the OS double-lock. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_non_context_synchronizing_exception_controls	int	0x1	Implement cosmetic controls for whether exception entry and exit are context synchronizing events (SCTLR_ELx.{EIS,EOS}) from ARMv8.5. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
has_par_bit10_razwi	bool	0x0	Whether PAR_EL1[10] is RAZ/WI
has_pointer_authentication	int	0x1	Implement ARMv8.3 pointer authentication. Possible values of this parameter are: - 1, feature is implemented if ARMv8.3 is enabled. - 2, feature is implemented.
has_prediction_invalidation_instructions	int	0x1	Implement execution and data prediction invalidation from ARMv8.5 Possible values of this parameter are: - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
has_pstate_dit	int	0x1	Implement timing insensitivity of data processing instructions from ARMv8.4. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_pstate_pan	int	0x1	Implement the PSTATE.PAN (Privileged Access Never) control from ARMv8.1 Possible values of this parameter are: - 1, feature is implemented if ARMv8.1 is enabled. - 2, feature is implemented.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-76 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
has_pstate_uao	int	0x1	Implement the PSTATE.UAO (User Access Override) control from ARMv8.2. Possible values of this parameter are: - 1, feature is implemented if ARMv8.2 is enabled. - 2, feature is implemented.
has_ras_armv84_extension	int	0x1	Implement ARMv8.4 RAS Extension. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_ras_critical_error	int	0x0	[DEPRECATED: Set CI field on first register in error_record_feature_register JSON instead] ARMv8.4 AArch64 RAS Critical Error is implemented or not. 0 - Feature Not Supported, 1 - Feature always enabled, 2 - Feature is controllable.
has_ras_double_fault	int	0x1	Implement ARMv8.4 RAS Doublt Fault Extension. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_ras_fault_injection	int	0x0	[DEPRECATED: Set INJ field on first register in error_record_feature_register JSON instead] Implement ARMv8.4 Standard Fault Injection mechanism. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_ras_timestamp	int	0x0	[DEPRECATED: Set TS field on first register in error_record_feature_register JSON instead] ARMv8.4 AArch64 RAS Timestamp register is implemented or not. 0 - No Timestamp is recorded, 1 - Generic Timer timestamp is recorded, 2 - IMP DEF timestamp is recorded.
has_restriction_on_speculative_data_loaded	int	0x1	Implements the ARMv8.5 security feature (Restrictions on the effects of speculation) Possible values of this parameter are: - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
has_rndr	int	0x0	Implement random number instructions to read from RNDR and RNDRSS random number registers from ARMv8.5. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
has_rounding_doubling_multiply_add_subtract	int	0x1	Implement the rounding doubling multiply add and subtract instructions from ARMv8.1 Possible values of this parameter are: - 1, feature is implemented if ARMv8.1 is enabled. - 2, feature is implemented.
has_secure_el2	int	0x1	Implement support for Secure EL2 Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_self_hosted_trace_extension	int	0x1	Implement support for the Self-hosted Trace Extensions from ARMv8.4. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-76 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
has_small_page_table	int	0x1	Implement small page table support which increases the maximum value of TxSZ field from ARMv8.4. Note: will be unimplemented only if both has_small_page_table=0x0 and has_secure_el2=0x0. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_speculation_barrier_inst	int	0x1	Implement speculation barrier instruction (SB) from ARMv8.5 Possible values of this parameter are: - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
has_stage2_fwb	int	0x1	Implement HCR_EL2.FWB, stage 2 control of memory types and cacheability Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_stage2_xnx	int	0x1	Implement the extended XN[1:0] stage 2 control from ARMv8.2. Possible values of this parameter are: - 1, feature is implemented if ARMv8.2 is enabled. - 2, feature is implemented.
has_stage2_xnx_in_aarch32	bool	0x1	Implement the extended XN[1:0] stage 2 control from ARMv8.2 for Aarch32.
has_statistical_profiling	bool	0x1	Whether Statistical Based Profiling is implemented
has_supersections	bool	0x1	Whether VMSAv8-32 supersection to support more than 32-bit PA using short descriptor is implemented.
has_tlb_conflict_abort	bool	0x0	Detected inconsistent TLB content generate aborts.
has_tlbi_range	int	0x1	Implement support for TLB Range Maintenance instructions (TLBI RVAE1, etc) from ARMv8.4. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_tlbi_to_outer_shareable	int	0x1	Implement support for TLB Maintenance instructions that extend to the Outer Shareable domain (TLBI VAE1OS, etc) from ARMv8.4. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_tlbi_ttl	int	0x1	Implement support for the TTL level hint in by-address TLB Maintenance instructions from ARMv8.4. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_unaligned_single_copy_atomicity	int	0x1	Implement support for SCTLR_ELx.nAA from ARMv8.4, and A64 atomic, exclusive and acquire/release instructions accessing unaligned bytes inside a 16byte window will not generate alignment fault. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_v8_4_debug_extension	int	0x1	Implement ARMv8.4 debug extensions Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-76 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
has_v8_4_pmu_extension	int	0x1	Implement PMU extension from ARMv8.4. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_v8_5_debug_over_power_down	int	0x0	Implement ARMv8.5 Debug over powerdown Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
has_v8_5_pmu_extension	int	0x1	Implement PMU extension from ARMv8.5. Possible values of this parameter are: - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
has_v8_5_spe_extension	int	0x1	Implement SPE extension from ARMv8.5 Possible values of this parameter are: - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
has_v8_6_pmu_events	int	0x1	Implements PMU events from ARMv8.6 Possible values of this parameter are: - 1, feature is implemented if ARMv8.6 is enabled. - 2, feature is implemented.
has_v8_7_fp_enhancements	int	0x1	Implements the Floating Point enhancements from Armv8.7 (introduces FPCR.FIZ/AH/NEP, etc.) Possible values of this parameter are: - 1, feature is implemented if Armv8.7 is enabled. - 2, feature is implemented.
has_v8_7_pmu_events	int	0x1	Implement PMU events from ARMv8.7. Possible values of this parameter are: - 1, feature is implemented if Armv8.7 is enabled. - 2, feature is implemented.
has_v8_7_pmu_extension	int	0x1	Implement PMU extension from ARMv8.7. Possible values of this parameter are: - 1, feature is implemented if Armv8.7 is enabled. - 2, feature is implemented.
has_v8_7_spe_extension	int	0x1	Implement SPE extension from ARMv8.7. Possible values of this parameter are: - 1, feature is implemented if Armv8.7 is enabled. - 2, feature is implemented.
has_vncr_el2	int	0x1	Implement support for nested virtualization enhancements from ARMv8.4. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_wfet_and_wfit	int	0x1	Implements WFE and WFI with Timeout from Armv8.7 Possible values of this parameter are: - 1, feature is implemented if Armv8.7 is enabled. - 2, feature is implemented.
has_writebuffer	bool	0x0	Implement write accesses buffering before L1 cache. May affect ext_abort behaviour.
has_xs	int	0x1	Implements Armv8.7 XS, TLBInXS, DSBnXS instruction Possible values of this parameter are: - 1, feature is implemented if Armv8.7 is enabled. - 2, feature is implemented.
icache-hit_latency	int	0x0	L1 I-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when icache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-76 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
icache-log2linelen	int	0x0	If nonzero, Log2 of the instruction cache line length in bytes (valid values in range 4-8). Otherwise the value of cache-log2linelen is used.
icache-maintenance_latency	int	0x0	L1 I-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when icache-state_modelled=true.
icache-miss_latency	int	0x0	L1 I-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when icache-state_modelled=true.
icache-nprefetch	int	0x1	Number of next sequential instruction cache lines to prefetch. This is only used when icache-prefetch_enabled=true.
icache-prefetch_enabled	bool	0x0	Enable simulation of instruction cache prefetching. This is only used when icache-state_modelled=true.
icache-prefetch_level	int	0x0	0 based cache level at which instructions are pre-fetched. This is only used when icache-prefetch_enabled=true.
icache-read_access_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per access (of size icache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if icache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when icache-state_modelled=true.
icache-read_bus_width_in_bytes	int	0x8	L1 I-Cache read bus width in bytes used to calculate per-access timing annotations
icache-read_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per byte accessed.icache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when icache-state_modelled=true.
icache-size	int	0x8000	L1 I-Cache size in bytes.
icache-ways	int	0x2	L1 I-Cache number of ways (sets are implicit from size).
ignore_DBGPRCR_CWRR	bool	0x0	Ignore writes to the deprecated DBGPRCR.CWRR bit.
ignore_access_flag_update_by_at_ops	bool	0x0	If true, AT operations do not update access flag
ignore_large_address_top_bits_in_page_walk	bool	0x0	Whether page table bits [15:12] are ignored if PA_SIZE < 52 and output address is configured < 52 with large page
imp_def_functionality_behaviour	int	0x0	Behaviour of IMPLEMENTATION DEFINED registers and system instructions. 0, UNDEF. 1, RAZ/WI.
impdef_regs_and_unpred_from_implementation	string	""	Configure implementation defined registers and unpredictable behaviour to match the specified implementation. Requires a license for the selected implementation model. Use ARM_Cortex-A<num> or ARM_<codename> for licensed pre-release cores.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-76 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
independent_cache_control_traps	int	0x0	Implement Independent Cache Control traps from ARMv8.5. 0, NO_SUPPORT. 1, SUPPORTED_BUT_NOT_FOR_TLB_MAINTENANCE_INSTRUCTIONS. 2, FULL_SUPPORT.
insert_iesb_before_exception	bool	0x0	If true then inserts an IESB before taking with Exception otherwise has no effect and IESB is taken after PState is changed due to the Exception.
instruction_tlb_size	int	0x0	Number of stage1+2 itlb entries (or 0 for unified ITLB+DTLB)
is_mt_res0	bool	0x0	If ARMv8.6 is not implemented, and PMUv3 is implemented, this parameter controls whether PMEVTYPER<n>.MT bit is RES0 or RW. For other implementations, this parameter has no effect
is_uniprocessor	bool	0x0	Value for the U bit in MPIDR. true disables L1 cache coherency protocols
l2cache-hit_latency	int	0x0	L2 Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
l2cache-maintenance_latency	int	0x0	L2 Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
l2cache-miss_latency	int	0x0	L2 Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
l2cache-read_access_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
l2cache-read_bus_width_in_bytes	int	0x8	L2 Cache read bus width in bytes used to calculate per-access timing annotations
l2cache-read_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per byte accessed.l2cache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
l2cache-size	int	0x80000	L2 Cache size in bytes.
l2cache-snoop_data_transfer_latency	int	0x0	L2 Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
l2cache-snoop_issue_latency	int	0x0	L2 Cache timing annotation latency for snoop accesses issued by this cache in total ticks. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-76 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
l2cache-ways	int	0x10	L2 Cache number of ways (sets are implicit from size).
l2cache-write_access_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-write_latency is set. This is only used when dcache-state_modelled=true.
l2cache-write_bus_width_in_bytes	int	0x8	L2 Cache write bus width in bytes used to calculate per-access timing annotations
l2cache-write_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per byte accessed. l2cache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
l3cache-mpamf.bwa_width_ns	int	0x10	L3 Cache width of MPAM bandwidth allocation fields for non-secure accesses.
l3cache-mpamf.bwa_width_s	int	0x10	L3 Cache width of MPAM bandwidth allocation fields for secure accesses.
l3cache-mpamf.cmax_width_ns	int	0x0	L3 Cache number of fractional bits in MPAM cache capacity partition control for non-secure accesses. Only the register interface is implemented - the control is NOT FUNCTIONAL.
l3cache-mpamf.cmax_width_s	int	0x0	L3 Cache number of fractional bits in MPAM cache capacity partition control for secure accesses. Only the register interface is implemented - the control is NOT FUNCTIONAL.
l3cache-mpamf.cpbm_width_ns	int	0x0	L3 Cache width of MPAM cache portion bitmap for non-secure accesses. If 0, the feature is not implemented, and all ways are available.
l3cache-mpamf.cpbm_width_s	int	0x0	L3 Cache width of MPAM cache portion bitmap for secure accesses. If 0, the feature is not implemented, and all ways are available.
l3cache-mpamf.csu_num_mon_ns	int	0x0	L3 Cache number of MPAM cache storage usage monitors for non-secure accesses.
l3cache-mpamf.csu_num_mon_s	int	0x0	L3 Cache number of MPAM cache storage usage monitors for secure accesses.
l3cache-mpamf.has_prop_ns	bool	0x0	Enable memory bandwidth proportional stride control for non-secure accesses. Only the register interface is implemented - the control is NOT FUNCTIONAL.
l3cache-mpamf.has_prop_s	bool	0x0	Enable memory bandwidth proportional stride control for secure accesses. Only the register interface is implemented - the control is NOT FUNCTIONAL.
max_32bit_el	int	0x3	Maximum exception level supporting AArch32 modes.
memory.acp.AxCACHE_mask	int	0x0	Used with memory.acp.AxCACHE_pattern to define which memory types the ACP port accepts. All transactions which do not satisfy (AxCACHE & mask) == pattern will abort.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-76 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
memory.acp.AxCACHE_pattern	int	0x0	Used with memory.acp.AxCACHE_mask to define which memory types the ACP port accepts. All transactions which do not satisfy (AxCACHE & mask) == pattern will abort.
memory.l2_cache.is_inner_cacheable	bool	0x1	L2 cache obeys inner cacheable attributes (rather than outer cacheable attributes)
memory.l2_cache.is_inner_shareable	bool	0x1	L2 cache obeys inner shareable attributes (rather than outer sharaable attributes)
memory_tagging_support_level	int	0x0	Specify the memory tagging extension support level: 0, not implemented. 1, instructions and registers only are implemented. 2, implemented. 3, implemented with asymmetric handling of exceptions
mixed_endian	int	0x1	Implement support for mixed endianness. 0, not supported. 1, supported at all exception levels. 2, supported at EL0 only.
mpam_force_ns_rao	bool	0x0	Whether MPAM3_EL3.FORCE_NS bit is RAO/WI.
mpam_frac	int	0x0	MPAM fractional revision number in ID_AA64PFR1_EL1.MPAM_frac field
mpam_has_hcr	bool	0x0	MPAM Whether MPAMIDR_EL1 HAS_HCR bit is set or clear
mpam_max_partid	int	0xffff	MPAM Maximum PARTID Supported
mpam_max_pmg	int	0xff	MPAM Maximum PMG Supported
mpam_max_vpmr	int	0x0	MPAM Maximum VPMR Supported
mpamidr_has_force_ns	int	0x0	Whether MPAMIDR_EL1.HAS_FORCE_NS bit is set or clear. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.6 is enabled. - 2, feature is implemented.
mpamidr_has_sdeflt	int	0x0	Whether MPAMIDR_EL1.HAS_SDEFLT bit is set or clear. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.6 is enabled. - 2, feature is implemented.
mpamidr_has_tidr	int	0x0	Whether MPAMIDR_EL1.HAS_TIDR bit is set or clear Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.6 is enabled. - 2, feature is implemented.
mvbar_reset_is_rvbar	bool	0x1	If true then the reset value of MVBAR is RVBAR, if false the reset value is UNKNOWN.
non_secure_vgic_alias_when_ns_only	int	0x0	If ! has_el3 and only non-secure side exists, then the normal position of the VGIC is a secure alias. If this parameter is non-zero then in addition a non-secure alias of the VGIC will be placed at this position (aligned to 32 KB).
num_loregion_descriptors	int	0x0	Number of Limited Ordering Region descriptors implemented (if ARM v8.1 extensions are implemented)
num_loregions	int	0x0	Number of Limited Ordering Regions implemented excluding background region (if ARM v8.1 extensions are implemented)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-76 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
number_of_error_records	int	0x0	Cores Number of Error records supported for RAS
optimal_alignment_size	int	0x1	Alignment boundary which does not incur additional performance penalty from ARMv8.5. - 1, architectural misalignment is used to set PMU event LDST_ALIGN_LAT and SPE event E[11] - 2, access crossing 4 byte boundary is used to set PMU event LDST_ALIGN_LAT and SPE event E[11] - 3, access crossing 8 byte boundary is used to set PMU event LDST_ALIGN_LAT and SPE event E[11] ... - 12, access crossing 4 Kbyte boundary is used to set PMU event LDST_ALIGN_LAT and SPE event E[11]
output_attributes	string	"ExtendedID[62:55]=MPAM_PMG, ExtendedID[54:39]=MPAM_PARTID, ExtendedID[38]=MPAM_NS"	User-defined transform to be applied to bus attributes like MasterID, ExtendedID or UserFlags. Currently, only works for MPAM Attributes encoding into bus attributes.
page_based_hardware_attributes	int	0x0	Implement the page based hardware attributes from ARMv8.2. This parameter indicates which page table bits are available for hardware, where bits[3:0] correspond to PTE[62:59] and to TCR_ELx.HWUnyy.
pmmir_el1_bus_slots	int	0x0	Largest value by which BUS_ACCESS can increment over BUS_CYCLES cycles. From v8.7 PMU extension
pmmir_el1_bus_width	int	0x0	Width, in bytes, of accesses counted by BUS_ACCESS. From v8.7 PMU extension
pseudo_fault_generation_feature_register	string	""	ARMv8.4 Standard Pseudo-fault generation feature register values. JSON schema for the parameter value is: [{"OF":false,"UC":false,"UEU":false,"UER":false,"UEO":false,"DE":false,"CE":0x0,"CI":false,"ER":false,"PN":false,"AV":false,"MV":false,"SYN":false,"R":false},other_psuedo-fault_generating_features_register_values]. Where OF, UC, UEU, UER, UEO, DE, CI, ER, PN, AV, MV, SYN, and R have valid false(NOT_SUPPORTED) and true(FEATURE_CONTROLLABLE), where CE can have 0(NOT_SUPPORTED), 1(NONSPECIFIC_CE_SUPPORTED) and 3(TRANSIENT_OR_PERSISTENT_CE_SUPPORTED). Effective only when ERXFR's INJ field allows it or has_ras_fault_injection is true.
pstate_ssbs_type	int	0x0	Implement speculative store bypass safe feature from ARMv8.5. 0, Not supported. 1, Supported without MSR/MRS access to SSBS. 2, fully supported.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-76 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
ptw_latency	int	0x0	Page table walker latency for TA (Timing Annotation), expressed in simulation ticks
randomize_unknowns_at_reset	bool	0x0	Will fill in unknown bits in registers at reset with random value using register_reset_data as seed, it overrides scramble_unknowns_at_reset
ras_errselr_undef_if_no_error_records	bool	0x0	If true, ERRSELR_EL1 will be undefined if ERRIDR_EL1 indicates that zero error records are implemented.
ras_frac	int	0x0	0, No additional feature implemented. 1, Additional ERXMISC*, ERXPFG* registers and FaultInjection trap from RAS v1.1. implemented
register_reset_data	int	0x0	Data used to fill register bits when they become UNKNOWN at reset.
restriction_on_speculative_execution	int	0x0	Implements the ARMv8.5 security feature (Restrictions on the effects of speculation): 0: No disclosure whether branch targets trained in one context can affect speculative execution in a different context, 1: Branch targets trained in one context cannot affect speculative execution in a different hardware described context (SCXTNUM_ELx not supported), 2: Branch targets trained in one context cannot affect speculative execution in a different hardware described context (SCXTNUM_ELx supported)
rmr_always_implemented	bool	0x0	Always implement RMR_ELx, RMR, or HRMR at the highest implemented exception level, even if that exception level cannot use both AArch32 and AArch64.
rndr_rndrrs_seed	int	0x0	Initial seed for random engine used in RNDR register
scheduler_mode	int	0x0	Control the interleaving of instructions in this processor. 0, default long quantum. 1, low latency mode, short quantum and signal checking. 2, lock-breaking mode, long quantum with additional context switches near load-exclusive instructions.
scramble_unknowns_at_reset	bool	0x1	Will fill in unknown bits in registers at reset with register_reset_data
set_rasv10_for_armv84_and_higher	bool	0x0	ARMv8.4 mandates RAS System Architecture v1.1, but when there are no error records and ARMv8.4-DFE is not implemented then there is no functional difference between the RAS System Architecture v1.0 (that is, the RAS extension as in pre-ARMv8.4 implementations) and the RAS System Architecture v1.1 (also known as ARMv8.4-RAS). This flag if true will set the RAS ID to declare RAS v1.0 rather than RAS v1.1 for ARMv8.4 and higher implementations. If this is set and the core does not conform to the restrictions then this parameter is ignored.
spsr_el3_is_mapped_to_spsr_mon	bool	0x1	Whether SPSR_EL3 is mapped to AArch32 register SPSR_mon
spsr_m4_res0	bool	0x0	Whether SPSR_ELx.M[4] bit should be RES0 for AARCH64 only implementations
stage12_tlb_size	int	0x80	Number of stage1+2 tlb entries.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-76 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
stage1_tlb_size	int	0x0	Number of stage1 only tlb entries.
stage1_walkcache_size	int	0x0	Number of stage1 only walk cache entries.
stage2_tlb_size	int	0x0	Number of stage2 only tlb entries.
stage2_walkcache_size	int	0x0	Number of stage2 only walk cache entries.
statistical_profiling_buffer_alignment	int	0x1	Statistical profiling alignment constraint for sample buffer
statistical_profiling_random_interval_is_separate	bool	0x0	Statistical profiling random interval gets added to the main timer interval(false) or (true) runs as separate timer
statistical_profiling_recommended_min_sampling	int	0x100	Statistical profiling recommended minimum sampling interval
take_ccfail_tsc_trap	bool	0x0	When take_ccfail_undef=1 this prameter controls whether or not an SMC instruction that is trapped by HCR_EL2.TSC but fails its condition code check generates a trap to EL2.
take_ccfail_undef	bool	0x1	UNDEF exception is taken even if condition code check fails
tcr_ps_reserved_value_size	int	0x0	Physical size treated when TCR.(I)PS is programmed with a reserved value. 0, 48 bits. 1, 52 bits. The parameter value is treated 0 if LPA is not supported.
tcr_txsz_undersize_should_fault	bool	0x1	If large VA is not supported, Whether undersized TxSZ value should generate translation fault
tdosa_traps_osdlr_if_no_os_double_lock	bool	0x1	MDCR_EL*.TDOSA enables trap on OSDLR_EL1 and DBGOSDLR when OS double-lock is not implemented.
tidcp_traps_el0_undef_imp_def	bool	0x1	TIDCP has priority over UNDEF for accesses to IMPLEMENTATION DEFINED functionality from EL0
tlb_latency	int	0x0	TLB latency for TA (Timing Annotation), expressed in simulation ticks
trace_physical_registers_when_host_virtualisation_enabled	int	0x0	When host virtualisation is enabled, trace sysreg accesses to physical register accessed (0=disabled, 1=Trace only ELR/SPSR_EL1 as ELR/SPSR_EL2, 2=Trace all redirected registers as physical registers
treat-dcache-cmos-to-pou-as-nop	bool	0x0	Whether dcache invalidation to the point of unification is required for instruction to data coherence. true - Invalidate operations not required
treat-dcache-invalidate-as-clean-invalidate	bool	0x0	Treat data cache invalidate operations as clean and invalidate.
undef_ccsidr2_access_for_unimplemented_aarch32	bool	0x0	Whether access to CCSIDR2 is undef if AArch32 is implemented or not
unification-level	int	0x1	Level of Unification Inner Shareable for the cache hierarchy
unification-uniprocessor-level	int	0x1	Level of Unification Uniprocessor for the cache hierarchy

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-76 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
unpred_nested_virtualization_nv_behaviour	int	0x0	Constrained unpredictable choices for HCR_EL2.NV=0 and HCR_EL2.NV1=1 with respect to nested virtualization - 0, Behave as defined in the sepcification as per bit values - 1, Behave as if HCR_EL2.NV=1 and HCR_EL2.NV1=1 for all purpose other than reading back HCR_EL2.NV - 2, Behave as if HCR_EL2.NV=0 and HCR_EL2.NV1=0 for all purpose other than reading back HCR_EL2.NV1
unpred_vncr_el2_ress_mismatch	int	0x0	Constrained unpredictable choices when bits marked as RESS do not all have the same value for VNCR_EL2 - 0, Generating an EL2 translation regime translation abort on use of the VNCR_EL2 register - 1, Reserved sign exteneded bits of VNCR_EL2 are same as bit[52] or bit[48] based on if large VA is supported or not, for all purposes other than reading back the register
unpred_zero_spsr_btype	bool	0x1	Constrained unpredictable control to make SPSR_ELx.BTYPE 0 instead of PSTATE.BTYPE on synchronous exceptions other than Software Step, PC alignment fault, Instruction Abort, Breakpoint or Address Matching Vector Catch, Illegal Execution State, BRK instruction, Branch Target
unpredictable_exclusive_abort_memtype	int	0x0	Cause MMU abort if exclusive access is not supported in certain memory type (0=exclusives allowed in all memory types, 1=exclusives abort in Device memory types, 2=exclusives abort in any type other than WB inner cacheable)
unpredictable_hvc_behaviour	int	0x0	HVC unpredictable behaviour. 0, UNDEF. 1, NOP.
unpredictable_smc_behaviour	int	0x0	SMC unpredictable behaviour. 0, UNDEF. 1, NOP.
use_Xt_as_LDG_STG_input	bool	0x1	Use new MTE Intructions formats for LDG/ST(Z)(2)G, which use Xt as an input
use_mte_eac_02_instructions_encoding	bool	0x1	Use new MTE Intructions encoding since MTE spec EAC 0.2 in ARMv8.5.
use_mte_eac_08_tfsr_encoding	bool	0x1	Use new MTE TFSR_ELx encodings since MTE spec EAC 0.8 in ARMv8.5.
use_rosetta_disass	int	0x1	Use Rosetta disassembly library. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
use_tlb_contig_hint	bool	0x0	Translation table entries with the contiguous hint bit set generate large TLB entries.
walk_cache_latency	int	0x0	Walk cache latency for TA (Timing Annotation), expressed in simulation ticks
warn_unpredictable_in_v7	bool	0x1	If true, behaviour which is unpredictable in V7 yet is predictable in V8 will produce a warning
watchpoint-log2secondary_restriction	int	0x0	log2 size of secondary restriction of FAR/EDWAR possible values on watchpoint hit for load/store operations.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-76 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
wnr_is_read_for_s2f_on_s1_atomic_instr_fault	bool	0x0	Whether WnR is 0 for stage2 fault on stage1 for atomic instructions
wnr_is_read_for_s2f_on_s1_dbm_update	bool	0x0	Whether WnR is 0 for stage2 fault on stage1 descriptor dbm update

Table 3-77 Parameters for ARMAEMv8-A_MP

Name	Type	Default value	Description
cpu0.CFGEND	bool	0x0	Endianness configuration at reset. 0, little endian. 1, big endian.
cpu0.CONFIG64	bool	0x1	Register width configuration at reset. 0, AArch32. 1, AArch64.
cpu0.CP15SDISABLE	bool	0x0	Initialize to disable access to some CP15 registers
cpu0.CP15SDISABLE2	bool	0x0	Initialize to disable access to some CP15 registers
cpu0.CRYPTODISABLE	bool	0x0	Disable cryptographic features.
cpu0.DCZID-log2-block-size	int	0x8	Log2 of the block size cleared by DC ZVA instruction (as read from DCZID_EL0).
cpu0.DCZVA_single_write	bool	0x0	Execute the DCZVA as a single write
cpu0.MPIDR-override	int	0x0	Override of MPIDR value. If nonzero will override the MT, cluster and CPU ID bits in MPIDR.
cpu0.RVBAR	int	0x0	Value of RVBAR_ELx register.
cpu0.RVBAR32	int	0x0	Reset vector address in AARCH32 when VINITHI is not set and ignore_rvbar_in_aarch32 is set
cpu0.SMPnAMP	bool	0x1	Enable broadcast messages necessary for correct SMP operation at reset.
cpu0.TEINIT	bool	0x0	Instruction set state when resetting into AArch32. 0, A32. 1, T32.
cpu0.VINITHI	bool	0x0	Reset value of SCTLR.V.
cpu0.ase-present	bool	0x1	Set whether the model has been built with NEON support
cpu0.clock_divider	int	0x1	Clock divider ratio for asymmetric MP clocking.
cpu0.clock_multiplier	int	0x1	Clock divider ratio for asymmetric MP clocking.
cpu0.crypto_aes	int	0x2	AES instructions supported (requires CryptoPlugin to be loaded). 0, not implemented. 1, AES instructions implemented. 2, AES and PMULL instructions implemented.
cpu0.crypto_sha1	int	0x1	SHA-1 instructions supported (requires CryptoPlugin to be loaded). 0, not implemented. 1, SHA1 instructions implemented.
cpu0.crypto_sha256	int	0x1	SHA-256 instructions supported (requires CryptoPlugin to be loaded). 0, not implemented. 1, SHA256 instructions implemented.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-77 Parameters for ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
cpu0.crypto_sha3	int	0x0	Implement ARMv8.4 SHA-3 instructions (requires CryptoPlugin to be loaded). Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
cpu0.crypto_sha512	int	0x0	Implement ARMv8.4 SHA-512 instructions (requires CryptoPlugin to be loaded). Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
cpu0.crypto_sm3	int	0x0	Implement ARMv8.4 SM-3 instructions (requires CryptoPlugin to be loaded). Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
cpu0.crypto_sm4	int	0x0	Implement ARMv8.4 SM-4 instructions (requires CryptoPlugin to be loaded). Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
cpu0.cti-intack_mask	int	0x1	Set bits represent that the corresponding trigger requires software acknowledge via CTIINTACK
cpu0.cti-number_of_claim_bits	int	0x0	Number of implemented bits in CTICLAIMSET
cpu0.cti-number_of_triggers	int	0x8	Number of cti event triggers (default: 8, valid values: {3, 8-32})
cpu0.enable_crc32	int	0x0	CRC32 instructions supported. 0, not implemented. 1, CRC32 instructions implemented.
cpu0.enable_trace_special_hlt_imm16	bool	0x0	Enable usage of parameter trace_special_hlt_imm16
cpu0.etm-present	bool	0x1	Set whether the model has ETM support
cpu0.force-fpsid	bool	0x0	Override the FPSID value
cpu0.force-fpsid-value	int	0x0	Value to override the FPSID value to
cpu0.has_hcptr_tase	bool	0x1	If false, HCPTR.TASE is RES0
cpu0.max_code_cache_mb	int	0x100	Maximum size of the simulation code cache (MiB). For platforms with more than 2 cores this limit will be scaled down. (e.g 1/8 for 16 or more cores)
cpu0.min_sync_level	int	0x0	Force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)
cpu0.number-of-breakpoints	int	0x10	Number of breakpoints.
cpu0.number-of-context-breakpoints	int	0x10	Number of breakpoints that are context aware.
cpu0.number-of-watchpoints	int	0x10	Number of watchpoints.
cpu0.operation_bandwidth	int	0x1	Operation width for ARMv8.4 PMU extension
cpu0.semihosting-A32_HLT	int	0xf000	A32 HLT number for semihosting calls.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-77 Parameters for ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
cpu0.semihosting-A64_HLT	int	0xf000	A64 HLT number for semihosting calls.
cpu0.semihosting-ARM_SVC	int	0x123456	A32 SVC number for semihosting calls.
cpu0.semihosting-T32_HLT	int	0x3c	T32 HLT number for semihosting calls.
cpu0.semihosting-Thumb_SVC	int	0xab	T32 SVC number for semihosting calls.
cpu0.semihosting-cmd_line	string	""	Command line available to semihosting calls.
cpu0.semihosting-cwd	string	""	Base directory for semihosting file access.
cpu0.semihosting-enable	bool	0x1	Enable semihosting SVC/HLT traps.
cpu0.semihosting-heap_base	int	0x0	Virtual address of heap base.
cpu0.semihosting-heap_limit	int	0xf000000	Virtual address of top of heap.
cpu0.semihosting-prefix	bool	0x0	Prefix semihosting output with target instance name
cpu0.semihosting-stack_base	int	0x10000000	Virtual address of base of descending stack.
cpu0.semihosting-stack_limit	int	0xf000000	Virtual address of stack limit.
cpu0.semihosting-stderr_istty	bool	0x1	Result for semihost istty call when argument is stderr
cpu0.semihosting-stdin_istty	bool	0x1	Result for semihost istty call when argument is stdin
cpu0.semihosting-stdout_istty	bool	0x1	Result for semihost istty call when argument is stdout
cpu0.semihosting-use_stderr	bool	0x0	Send stderr from the simulated process to host stderr
cpu0.trace_special_hlt_imm16	int	0xf000	For this HLT number, IF enable_trace_special_hlt_imm16=true, skip performing usual HLT execution but call MTI trace if registered
cpu0.unpredictable_WPMASKANDBAS	int	0x1	Constrained unpredictable handling of watchpoints when mask and BAS fields specified. 0, IGNOREMASK. 1, IGNOREBAS (default). 2, REPEATBAS8. 3, REPEATBAS.
cpu0.vfp-enable_at_reset	bool	0x0	Enable VFP in CPACR, CPPWR, NSACR at reset. Warning: Arm recommends going through the implementation's suggested VFP power-up sequence!
cpu0.vfp-present	bool	0x1	Set whether the model has VFP support
cpu0.vfp-traps	bool	0x1	Implement support for trapping floating-point exceptions
cpu0.vfp-traps-show-all	bool	0x0	Report all trapped floating-point exceptions in the syndrome when a combination occurs.
cpu0.wfet_early_or_delayed_timeout	int	0x0	WFET early or delayed timeout beyond the threshold value of CNTVCT_EL0 in percentage.
cpu0.wfit_early_or_delayed_timeout	int	0x0	WFIT early or delayed timeout beyond the threshold value of CNTVCT_EL0 in percentage.

3.4.3 ARMAEMv8A_AEMv8AMPCT

ARM AEM v8-A+v8-A (MP) CPU component - number of core configurable at runtime. This model is written in C++.

ARMAEMv8A_AEMv8AMPCT contains the following CADI targets:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

- ARMAEMv8-A_MP
- Cluster_ARMAEMv8-A_MP
- Subcluster_ARMAEMv8-A_MP

ARMAEMv8A_AEMv8AMPCT contains the following MTI components:

- *4.4 ARM_AEMv8-A_MP - trace* on page 4-908
- *4.48 ARMv8Cluster - trace* on page 4-2401
- *4.68 DSU - trace* on page 4-2463
- *4.71 GICv2 - trace* on page 4-2470
- *4.73 GICv3CPUInterfaceDecoder - trace* on page 4-2489
- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.108 PVCache - trace* on page 4-2761
- *4.114 TLB - trace* on page 4-2834

About ARMAEMv8A_AEMv8AMPCT

- AEMv8-A is deprecated and will be replaced by AEMvA, which enables both the existing Armv8-A architecture and Future Architecture Technologies. Support for architectural features is selectable through parameters. For additional information, contact *support@arm.com*.
- This component supports multiple versions of the Armv8-A architecture and provides parameters to enable or disable particular architectural features. Some of these parameters allow you to enable features in versions of the architecture earlier than the one in which they were introduced. These parameters have the possible values:

  0 Feature is not enabled.

  1 Feature is enabled, but only if the model is set to implement a version of the architecture in which the feature is supported.

  2 Feature is enabled, regardless of which architecture version the model implements.
- Each subcluster can be configured independently by using the parameters prefixed with `subcluster`*X* where *X* is 0-3.
- The number of cores is configured using the `NUM_CORES` parameter. The per-core parameters are preceded by `cpu`*n*, where *n* identifies the core (0-7).

AEMv8A core personalities

The AEM core personalities feature allows you to configure AEM instances in a platform to use the IMPLEMENTATION DEFINED registers and UNPREDICTABLE behavior for a specific implementation of your choice at model startup.

Configuring an AEM with a core personality requires a license for both the AEM and the selected implementation.

Set the personality using the environment variable `FASTSIM_AEMV8_PROFILE`, or the parameter `impdef_regs_and_unpred_from_implementation`.

The parameter allows you to configure different instances in the same platform with different personalities, including subclusters in a heterogeneous AEM. The environment variable takes precedence over the parameter and affects all instances of the AEM in the platform. Otherwise the two options function identically.

To see the available values for the environment variable or parameter, set either of them to the special value `list`. The model prints the list of available values and exits. An example value is `ARM_Cortex-A57`. Then set the parameter or environment variable to the required value.

Configuring a core personality only affects IMPLEMENTATION DEFINED registers and UNPREDICTABLE behavior. In other respects, the cluster or subcluster still behaves like the AEM. For example, all parameters default to the AEM default values. Therefore, parameters have to be manually

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

configured to valid values for the configured personality, as many of the defaults are incompatible with any given implementation.

To assist with this configuration, the AEM prints warnings for any parameter with an invalid value for the configured personality. The warning lists the parameter name and the valid value or range of values it can be set to for the selected personality.

Running the model with invalid parameter configurations for the selected personality can lead to unexpected behavior.

Mapping architectural features to Fast Models releases

The following table shows which Fast Models releases the key architectural features are implemented in, from Armv8.3-A onwards. Support is at EAC quality unless otherwise stated.

Table 3-78 Architectural features and Fast Models releases

Architecture version	Feature	Fast Models version									
		11.13	11.12	11.11	11.10	11.9	11.8	11.7	11.6	11.5	11.4
Armv8.3-A	All [n]	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Armv8.4-A	All [o]	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Armv8.5-A	Memory Tagging Extension (MTE) functionality	✓	✓	✓	✓	✓	✓	✓	✓ (Beta)	-	-
	MTE performance	✓	✓	✓	✓	✓	✓ (Beta)	-	-	-	-
	All except MTE [p].	✓	✓	✓	✓	✓	✓	✓	✓	✓	-
Armv8.6-A	All [q]	✓	✓	✓	✓	✓	-	-	-	-	-
Armv8.7-A	Limited TLBI maintenance	✓ (Alpha)	-	-	-	-	-	-	-	-	-
	All except TLBI maintenance [r]	✓	-	-	-	-	-	-	-	-	-
Future Architecture Technologies	SVE2, TME, CSRE, BRBE. [s]	✓	✓	✓	✓	✓	✓	-	-	-	-

[n] Armv8.3-A includes pointer authentication, nested virtualization, Advanced SIMD complex number support, improved Javascript data type conversion support, memory consistency model changes, support for larger system-visible caches, debug over powerdown.

[o] Armv8.4-A includes Secure EL2, additional cryptographic hashes, activity monitors, enhanced support for nested virtualization, Memory Partitioning and Monitoring (MPAM), stage 2 forced write-back, dot product instructions, data independent timing instructions, Large System Extensions (LSE), support for TLB maintenance instructions and for TLB range instructions, Translation Table Level (TTL), enhancements to weaker release consistency, debug relaxations and extensions, flag manipulation instructions.

[p] Armv8.5-A includes Branch Target Indicators (BTI), Enhanced Virtualization Traps (EVT), Random Number Generator (RNG), Flag Manipulation instructions, Floating-point round to integer, DC CVADP.

[q] Armv8.6-A includes General Matrix Multiply (GEMM), EnhancedPAC2, FPAC, Data Gathering Hint, Additional PMU events, Enhanced Counter Virtualization.

[r] Armv8.7-A includes WFE and WFI with timeouts, Larger physical address for 4KB and 16KB translation granules, MTE Asymmetric Fault Handling, Enhancements to Privilege Access Never (PAN), Enhancements to PMU and SPE.

[s] Implemented by AEMvA only.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-79 Ports

Name	Protocol	Type	Description
CNTHPIRQ[8]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTHPSIRQ[8]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTHVIRQ[8]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTHVSIRQ[8]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTPNSIRQ[8]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTPSIRQ[8]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTVIRQ[8]	*Signal* on page 2-85	Master	Timer signals to SOC.
CRITICALIRQ[8]	*Signal* on page 2-85	Master	RAS Critical Error Interrupt.
ERRORIRQ[8]	*Signal* on page 2-85	Master	RAS Error Recovery Interrupt.
FAULTIRQ[8]	*Signal* on page 2-85	Master	RAS Fault Handling Interrupt
acp_s	PVBus	Slave	AXI ACP slave port.
broadcastatomic	*Signal* on page 2-85	Slave	CHI defined pins.
broadcastcachemaint	*Signal* on page 2-85	Slave	ACE defined pins.
broadcastinner	*Signal* on page 2-85	Slave	ACE defined pins.
broadcastouter	*Signal* on page 2-85	Slave	ACE defined pins.
broadcastpersist	*Signal* on page 2-85	Slave	CHI defined pins.
cfgend[8]	*Signal* on page 2-85	Slave	This signal if for EE bit initialisation.
cfgsdisable	*Signal* on page 2-85	Slave	This signal disables write access to some secure Interrupt Controller registers.
clk_in	ClockSignal	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the cluster level components run e.g cluster level timers, caches and pmu.
clrexmonack	*Signal* on page 2-85	Master	Acknowledge handshake signal for the clrexmonreq signal
clrexmonreq	*Signal* on page 2-85	Slave	Signals the clearing of an external global exclusive monitor
clusterid	*Value* on page 2-85	Slave	The port reads the value in CPU ID register field, bits[11:8] of the MPIDR.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-79 Ports (continued)

Name	Protocol	Type	Description
cntvalueb	*CounterInterface* on page 2-84	Slave	Interface to SoC level counter module.
commirq[8]	*Signal* on page 2-85	Master	Interrupt signal from debug communications channel.
config64[8]	*Signal* on page 2-85	Slave	Register width after reset.
core_clk_in[8]	ClockSignal	Slave	The clock signal connected to the core_clk_in port is used to determine the rate at which each core executes instructions.
cp15sdisable[8]	*Signal* on page 2-85	Slave	This signal disables write access to some system control processor registers.
cpuporeset[8]	*Signal* on page 2-85	Slave	CPU power on reset. Initializes all the processor logic, including debug logic.
cryptodisable[8]	*Signal* on page 2-85	Slave	Disable cryptography extensions after reset.
cti[8]	*v8EmbeddedCrossTrigger_controlprotocol* on page 2-84	Master	Cross trigger matrix port.
cti0extin[4]	*Signal* on page 2-85	Slave	CTI trace inputs for core 0.
cti0extout[4]	*Signal* on page 2-85	Master	CTI trace outputs for core 0.
cti1extin[4]	*Signal* on page 2-85	Slave	CTI trace inputs for core 1.
cti1extout[4]	*Signal* on page 2-85	Master	CTI trace outputs for core 1.
cti2extin[4]	*Signal* on page 2-85	Slave	CTI trace inputs for core 2.
cti2extout[4]	*Signal* on page 2-85	Master	CTI trace outputs for core 2.
cti3extin[4]	*Signal* on page 2-85	Slave	CTI trace inputs for core 3.
cti3extout[4]	*Signal* on page 2-85	Master	CTI trace outputs for core 3.
cti4extin[4]	*Signal* on page 2-85	Slave	CTI trace inputs for core 4.
cti4extout[4]	*Signal* on page 2-85	Master	CTI trace outputs for core 4.
cti5extin[4]	*Signal* on page 2-85	Slave	CTI trace inputs for core 5.
cti5extout[4]	*Signal* on page 2-85	Master	CTI trace outputs for core 5.
cti6extin[4]	*Signal* on page 2-85	Slave	CTI trace inputs for core 6.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-79 Ports (continued)

Name	Protocol	Type	Description
cti6extout[4]	*Signal* on page 2-85	Master	CTI trace outputs for core 6.
cti7extin[4]	*Signal* on page 2-85	Slave	CTI trace inputs for core 7.
cti7extout[4]	*Signal* on page 2-85	Master	CTI trace outputs for core 7.
ctidbgirq[8]	*Signal* on page 2-85	Master	Cross Trigger Interface (CTI) interrupt trigger output.
dbgen[8]	*Signal* on page 2-85	Slave	External debug interface.
dbgnopwrdwn[8]	*Signal* on page 2-85	Master	Debug no power down request.
dbgpwrdwnack[8]	*Signal* on page 2-85	Master	Debug power down acknowledge.
dbgpwrdwnreq[8]	*Signal* on page 2-85	Slave	Debug power down request.
dbgpwrupreq[8]	*Signal* on page 2-85	Master	Debug power up request.
dev_debug_s	PVBus	Slave	External debug interface.
etm[8]	PVBus	Master	Embedded trace macrocell port.
event	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE.
external_trace_reset[8]	*Signal* on page 2-85	Slave	ETMv4 External Trace Reset signal.
fiq[8]	*Signal* on page 2-85	Slave	This signal drives the CPUs fast-interrupt handling.
gicv3_redistributor_s[8]	*GICv3Comms* on page 2-84	Slave	GICv3 cpu interface ports.
irq[8]	*Signal* on page 2-85	Slave	This signal drives the CPUs interrupt handling.
irqs[224]	*Signal* on page 2-85	Slave	These signals drive the CPU's interrupt controller interrupt lines.
l2reset	*Signal* on page 2-85	Slave	This signal resets timer and interrupt controller.
memorymapped_debug_s	PVBus	Slave	External debug interface.
niden[8]	*Signal* on page 2-85	Slave	External debug interface.
pchannel_cluster	*PChannel* on page 2-80	Slave	PChannel for cluster.
pchannel_core[8]	*PChannel* on page 2-80	Slave	PChannels for cores
periphbase	*Value_64* on page 2-85	Slave	This port sets the base address of private peripheral region.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-79 Ports (continued)

Name	Protocol	Type	Description
pmbirq[8]	*Signal* on page 2-85	Master	Interrupt signal from the statistical profiling unit.
pmuirq[8]	*Signal* on page 2-85	Master	Interrupt signal from performance monitoring unit.
presetdbg[8]	*Signal* on page 2-85	Slave	Initialize the shared debug APB, Cross Trigger Interface (CTI), and Cross Trigger Matrix (CTM) logic.
pvbus_m0	PVBus	Master	The core will generate bus requests on this port.
rei[8]	*Signal* on page 2-85	Slave	Individual processor RAM Error Interrupt signal input.
reset[8]	*Signal* on page 2-85	Slave	Raising this signal will put the core into reset mode.
romaddr	*Value_64* on page 2-85	Slave	Debug ROM base address.
romaddrv	*Signal* on page 2-85	Slave	Debug ROM base address valid.
rvbar[8]	*Value_64* on page 2-85	Slave	Reset vector base address.
sei[8]	*Signal* on page 2-85	Slave	Per core System Error physical pins.
smpnamp[8]	*Signal* on page 2-85	Master	This signals AMP or SMP mode for each core.
spiden[8]	*Signal* on page 2-85	Slave	External debug interface.
spniden[8]	*Signal* on page 2-85	Slave	External debug interface.
standbywfe[8]	*Signal* on page 2-85	Master	This signal indicates if a core is in WFE state.
standbywfi[8]	*Signal* on page 2-85	Master	This signal indicates if a core is in WFI state.
standbywfil2	*Signal* on page 2-85	Master	This signal indicated all cores and L2 are in a power down state
teinit[8]	*Signal* on page 2-85	Slave	This signal provides default exception handling state.
ticks[8]	*InstructionCount* on page 2-84	Master	This port should be connected to one of the two ticks ports on a 'visualisation' component, in order to display a running instruction count.
trace_unit_reset[8]	*Signal* on page 2-85	Slave	ETMv4 Trace Unit Reset signal.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-79 Ports (continued)

Name	Protocol	Type	Description
vcpumntirq[8]	*Signal* on page 2-85	Master	Interrupt signal for virtual CPU maintenance IRQ.
vfiq[8]	*Signal* on page 2-85	Slave	Virtual FIQ input. Note that the irq/fiq pins are wired directly to the core if there is no internal VGIC. If there is an internal VGIC then these are ignored.
vinithi[8]	*Signal* on page 2-85	Slave	This signal controls of the location of the exception vectors at reset.
virq[8]	*Signal* on page 2-85	Slave	Virtual IRQ input. Note that the irq/fiq pins are wired directly to the core if there is no internal VGIC. If there is an internal VGIC then these are ignored.
virtio_s	PVBus	Slave	The virtio coherent port, hooks directly into the L2 system and becomes coherent (assuming attributes are set correctly).
vsei[8]	*Signal* on page 2-85	Slave	Processor Virtual System Error Interrupt request.

Table 3-80 Parameters for Subcluster_ARMAEMv8-A_MP

Name	Type	Default value	Description
subcluster0.ADFSR-AIFSR-implemented	bool	0x0	ADFSR and AIFSR are implemented
subcluster0.AIDR	int	0x0	Value of AIDR_EL1 register.
subcluster0.AMPIDR	int	0x400bbb000	Value of AMU Peripheral Identification Register
subcluster0.BPIMVA_causes_translation_lookup	bool	0x0	Do a translation when BPIMVA instruction is executed (which may cause a translation fault).
subcluster0.CCSIDR-L1D_override	int	0x0	If nonzero, override the value presented in CCSIDR for L1D (this is cosmetic and does not affect cache behaviour).
subcluster0.CCSIDR-L1I_override	int	0x0	If nonzero, override the value presented in CCSIDR for L1I (this is cosmetic and does not affect cache behaviour).
subcluster0.CCSIDR-L2_override	int	0x0	If nonzero, override the value presented in CCSIDR for L2 (this is cosmetic and does not affect cache behaviour).
subcluster0.CCSIDR-L3_override	int	0x0	If nonzero, allow L3 selection in CSSELR and present this value in CCSIDR (this is cosmetic and does not affect cache behaviour).
subcluster0.CPUCFR	int	0x0	Value of CPU Configuration Register

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-80 Parameters for Subcluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
subcluster0.CTIPIDR	int	0x0	If non-zero, override the CTI Peripheral Identification Register
subcluster0.DBGPIDR	int	0x0	If non-zero, override the Debug Peripheral Identification Register
subcluster0.ERXMISC0_mask	int	0x0	Write Mask for ERXMISC0 RAS Register
subcluster0.GMID-log2-block-size	int	0x4	Log2 of the block size accessed by STGM/LDGM/STZGM instructions
subcluster0.ISV_set_to_0_for_stage2_synch_external_abort	bool	0x0	Whether ESR_EL2.ISV is set to 0 on stage 2 synchronous external aborts
subcluster0.MIDR	int	0x410fd0f0	Value of MIDR_EL1 register.
subcluster0.NUM_CORES	int	0x1	Number of cores per cluster
subcluster0.PA_SIZE	int	0x28	Physical address range supported. For ARMv8.0 and ARMv8.1 this is limited to 48 bits.
subcluster0.PMCEID0	int	0xffffffff	Performance Monitor Common Event ID Reg 0 value - 64 bit
subcluster0.PMCEID1	int	0xffffffff	Performance Monitor Common Event ID Reg 1 value - 64 bit
subcluster0.PMUPIDR	int	0x0	If non-zero, override the PMU Peripheral Identification Register
subcluster0.abort_execution_from_device_memory	bool	0x0	Execution from device memory generates a prefetch abort.
subcluster0.advsimd_bf16_support_level	int	0x0	Implement BFloat16 operations from ARMv8.6. AArch64 Advanced SIMD and FP BFloat16 instructions are automatically enabled when has_arm_v8-6 is true. - 0, Not implemented. - 1, AArch64 Advanced SIMD and FP BFloat16 instructions only. - 2, AArch32 Advanced SIMD and VFP BFloat16 instructions only. - 3, Both AArch64 Advanced SIMD and FP and AArch32 Advanced SIMD and VFP BFloat16 instructions.
subcluster0.advsimd_i8mm_support_level	int	0x0	Implement Int8 matrix multiply operations from ARMv8.6. AArch64 Advanced SIMD and FP Int8 matrix multiply instructions are automatically enabled when has_arm_v8-6 is true. - 0, Not implemented. - 1, AArch64 Advanced SIMD and FP Int8 matrix multiply instructions only. - 2, AArch32 Advanced SIMD and VFP Int8 matrix multiply instructions only. - 3, Both AArch64 Advanced SIMD and FP and AArch32 Advanced SIMD and VFP Int8 matrix multiply instructions.
subcluster0.amair_reg_rw_mask	int	0x0	RW mask for implementation-defined registers.
subcluster0.amu_aux_type_fixed	string	""	Lists which AMU auxiliary registers that are fixed and to which event type. The JSON schema is: {fixed_aux_reg:evt_type, ...}. For example {"0":0x300} would make auxiliary register 0 fixed to event type 0x300
subcluster0.amu_aux_voffset_mask	int	0x0	If ARMv8.6 is implemented, each bit of the field, 0 to 15, when 1 indicates that the corresponding virtual offset register, AMEVCNTVOFF1<n>_EL2, is implemented

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-80 Parameters for Subcluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
subcluster0.amu_has_external_interface	int	0x0	Implement external memory-mapped access to system register of activity monitor unit from ARMv8.4. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
subcluster0.amu_has_sysreg_interface	bool	0x1	Implement system register access to activity monitor unit from ARMv8.4.
subcluster0.amu_num_auxiliary_counters	int	0x0	Number of AMU auxiliary counters implemented
subcluster0.amu_reset_domain	int	0x0	Reset domain for activity monitor unit. 0, COLD_RESET. 1, WARM_RESET. 2, NONE.
subcluster0.amu_version	int	0x1	Selects the activity monitor version implemented - 1, AMUv1 for Armv8.4 is implemented. - 2, AMUv1 for Armv8.6 is implemented.
subcluster0.apsr_read_restrict	bool	0x0	At EL0, unknown bits of APSR are RAZ.
subcluster0.arm_v8_7_accelerator_support_level	int	0x0	Implements Armv8.7 accelerator support instructions LD/ST64B, ST64BV, ST64BV0, etc.
subcluster0.auxilliary_feature_register0	int	0x0	Value of AFR0 ID register.
subcluster0.branch-predictor-clear-policy	int	0x2	Set branch prediction policy as defined for MMFR1[31:28]. This does not change the behaviour of the branch predictor, only what is reported in MMFR1.BPred.
subcluster0.branch-predictor-supported-ops	int	0x1	Set branch prediction policy as defined for MMFR3[11:8]. This does not change the behaviour of the branch predictor, only what is reported in MMFR3.BPMaint.
subcluster0.cache-log2linelen	int	0x6	Log2 of the cache line length in bytes.
subcluster0.cache_maintenance_hits_watchpoints	bool	0x0	DCIMVA operations executed in AArch32 modes hit watchpoints.
subcluster0.changing_block_size_without_bbm_support	int	0x0	Level of support for changing block size without break-before-make.
subcluster0.clear_IT_when_IL_set	bool	0x0	Clear IT bits when performing a *legal* exception return to AArch32 when IL is set
subcluster0.clear_IT_when_IL_set_explicitly	bool	0x0	Apart from clear_IT_when_IL_set, also clear IT bits when loading CPSR from SPSR/memory and IL == 1 in the value being loaded.
subcluster0.clear_reg_top_eret	int	0x1	Behaviour of the upper 32-bits of the Xn registers when changing between AArch32 state and AArch64 state. 0, upper 32-bits preserved for all registers. 1, upper 32-bits set to 0 for all accessible registers. 2, upper 32-bits set to 0 for a random selection of accessible registers. 3, upper-32-bits set to 0 for registers touched in AArch32
subcluster0.cpi_div	int	0x1	Divider for calculating CPI (Cycles Per Instruction)
subcluster0.cpi_mul	int	0x1	Multiplier for calculating CPI (Cycles Per Instruction)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-80 Parameters for Subcluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
subcluster0.dc_fault_unaligned_s1_device_s2_fwb	bool	0x0	Whether takes an Alignment Fault caused by the memory type on a DC {ZVA,GZVA,GVA} if the stage 1 memory type is any Device memory type
subcluster0.dcache-hit_latency	int	0x0	L1 D-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
subcluster0.dcache-maintenance_latency	int	0x0	L1 D-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
subcluster0.dcache-miss_latency	int	0x0	L1 D-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
subcluster0.dcache-prefetch_enabled	bool	0x0	Enable simulation of data cache prefetching. This is only used when dcache-state_modelled=true
subcluster0.dcache-read_access_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per access (of size dcache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
subcluster0.dcache-read_bus_width_in_bytes	int	0x8	L1 D-Cache read bus width in bytes used to calculate per-access timing annotations
subcluster0.dcache-read_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per byte accessed.dcache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
subcluster0.dcache-size	int	0x8000	L1 D-Cache size in bytes.
subcluster0.dcache-snoop_data_transfer_latency	int	0x0	L1 D-Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
subcluster0.dcache-ways	int	0x2	L1 D-Cache number of ways (sets are implicit from size).
subcluster0.dcache-write_access_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per access (of size dcache-write_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-write_latency is set. This is only used when dcache-state_modelled=true.
subcluster0.dcache-write_bus_width_in_bytes	int	0x8	L1 D-Cache write bus width in bytes used to calculate per-access timing annotations

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-80 Parameters for Subcluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
subcluster0.dcache-write_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per byte accessed. dcache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
subcluster0.dcimva_requires_s2_write_permissions	bool	0x0	Data-cache invalidate by MVA operations require stage 2 write permission (virtualised AArch32 guest).
subcluster0.dczva_reports_lowest_addr_on_tag_check_fail	bool	0x0	Whether DC ZVA reports lowest address in FAR on tag check fail
subcluster0.delay_serror	int	0x0	Add a propagation delay of serror signal into the core
subcluster0.delayed_dbgreg_between_secure_views	int	0x1	If delayed_dbgreg is enabled, whether the secure and nonsecure external views require explicit synchronization. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
subcluster0.dic-spi_count	int	0x40	Number of shared peripheral interrupts implemented
subcluster0.disable_sve_plugin	bool	0x0	If true, SVE will not be implemented in this processor even if the plugin is loaded
subcluster0.disable_unknown_update_event_on_reset	bool	0x0	Disables SYSREG_UPDATE event notification on reset for the registers whose bitfields are all reserverd or resets to architecturally unknown value
subcluster0.ecv_support_level	int	0x0	Implement Enhanced Counter Virtualization feature from ARMv8.6. 0, Not supported. 1, fully supported without CNTPOFF. 2, fully supported with CNTPOFF.
subcluster0.el0_can_access_imp_def_functionality	bool	0x0	If not made UNDEF by imp_def_functionality_behaviour, EL0 can access IMPLEMENTATION DEFINED registers and system instructions.
subcluster0.el0_el1_only_non_secure	bool	0x0	Secure/non-secure state if EL2 and EL3 are not implemented. 0, secure. 1, non-secure.
subcluster0.enable_tlb_contig_check	bool	0x0	Perform extra pagetable walks to check translation table entries that have the contiguous hint bit set.
subcluster0.enhanced_pac2_level	int	0x0	Implements Enhanced PAC2 from ARMv8.6. This feature is mandatory for ARMv8.6 but can be cherrypicked to a ARMv8.3(or greater) implementation. 0: No EnhancedPAC2, 1: EnhancedPAC2 Only, 2: EnhancedPAC2 with FPAC, 3: EnhancedPAC2 with FPACCombined.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-80 Parameters for Subcluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
subcluster0.error_record_feature_register	string	""	RAS feature register values. An array of JSON objects. The JSON schema for the array is: [{"ED":0x0,"IMPDEF_3_2":0x0,"UI":0x0,"FI":0x0,"UE":0x0,"CFI":0x0,"CEC":0x0,"RP":0x0,"DUI":0x0,"CEO":0x0,"CI":0x0,"TS":0x0,"INJ":0x0,"FRX":0x0,"UC":0x0,"UEU":0x0,"UER":0x0,"UEO":0x0,"DE":0x0,"CE":0x0,"Visibility":"Core"},other_feature_register_values]. Where ED,UI,FI,CE and UE have valid values betwn 0x0 - 0x3. CFI and DUI have valid values 0x0, 0x2 and 0x3. CEC has valid values 0x0,0x2 or 0x4. RP,CEO,INJ,FRX,UC,UEU,UER,UEO,DE has valid values 0x0 or 0x1. CI and TS has valid values of 0x0, 0x1 and 0x2. Visibility has valid values "Core" or "Cluster"
subcluster0.error_record_feature_register_json_file	string	""	File path to the RAS feature register values as JSON. The file uses the same format as the error_record_feature_register parameter value
subcluster0.exercise_stxr_fail	bool	0x0	Reject a pseudo-random majority of exclusive store instructions
subcluster0.ext_abort_device_GRE_prefetch_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_index, Valid indices in range [0, number_of_error_records).
subcluster0.ext_abort_device_GRE_prefetch_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
subcluster0.ext_abort_device_GRE_read_is_critical	bool	0x0	Critical reporting of device-GRE read external aborts.
subcluster0.ext_abort_device_GRE_read_is_sync	int	0x2	Synchronous reporting of device-GRE read external aborts. 0, asynchronous. 1, synchronous. 2, same as ext_abort_device_read_is_sync.
subcluster0.ext_abort_device_GRE_read_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_device_read_ras_index, Valid indices in range [0, number_of_error_records).
subcluster0.ext_abort_device_GRE_read_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_device_read_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
subcluster0.ext_abort_device_GRE_write_is_critical	bool	0x0	Critical reporting of device-GRE write external aborts
subcluster0.ext_abort_device_GRE_write_is_sync	int	0x2	Synchronous reporting of device-GRE write external aborts. 0, asynchronous. 1, synchronous. 2, same as ext_abort_device_write_is_sync.
subcluster0.ext_abort_device_GRE_write_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_device_write_ras_index, Valid indices in range [0, number_of_error_records).
subcluster0.ext_abort_device_GRE_write_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_device_write_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-80 Parameters for Subcluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
subcluster0.ext_abort_device_nGRE_prefetch_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_index, Valid indices in range [0, number_of_error_records).
subcluster0.ext_abort_device_nGRE_prefetch_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
subcluster0.ext_abort_device_nGRE_read_is_critical	bool	0x0	Critical reporting of device-nGRE read external aborts.
subcluster0.ext_abort_device_nGRE_read_is_sync	int	0x2	Synchronous reporting of device-nGRE read external aborts. 0, asynchronous. 1, synchronous. 2, same as ext_abort_device_read_is_sync.
subcluster0.ext_abort_device_nGRE_read_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_device_read_ras_index, Valid indices in range [0, number_of_error_records).
subcluster0.ext_abort_device_nGRE_read_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_device_read_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
subcluster0.ext_abort_device_nGRE_write_is_critical	bool	0x0	Critical reporting of device-nGRE write external aborts
subcluster0.ext_abort_device_nGRE_write_is_sync	int	0x2	Synchronous reporting of device-nGRE write external aborts. 0, asynchronous. 1, synchronous. 2, same as ext_abort_device_write_is_sync.
subcluster0.ext_abort_device_nGRE_write_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_device_write_ras_index, Valid indices in range [0, number_of_error_records).
subcluster0.ext_abort_device_nGRE_write_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_device_write_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
subcluster0.ext_abort_device_prefetch_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_index, Valid indices in range [0, number_of_error_records).
subcluster0.ext_abort_device_prefetch_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
subcluster0.ext_abort_device_read_is_critical	bool	0x0	Critical reporting of device-nGnRE read external aborts
subcluster0.ext_abort_device_read_is_sync	bool	0x1	Synchronous reporting of device-nGnRE read external aborts
subcluster0.ext_abort_device_read_ras_index	int	0x0	External Aborts are reported in RAS record index specified in this param. Values: Valid indices in range [0, number_of_error_records).

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-80 Parameters for Subcluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
subcluster0.ext_abort_device_read_ras_type	int	0x0	External Aborts are reported as RAS error type specified in this param. Values: 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
subcluster0.ext_abort_device_write_is_critical	bool	0x0	Critical reporting of device-nGnRE write external aborts
subcluster0.ext_abort_device_write_is_sync	bool	0x0	Synchronous reporting of device-nGnRE write external aborts
subcluster0.ext_abort_device_write_ras_index	int	0x0	External Aborts are reported in RAS record index specified in this param. Values: Valid indices in range [0, number_of_error_records).
subcluster0.ext_abort_device_write_ras_type	int	0x0	External Aborts are reported as RAS error type specified in this param. Values: 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
subcluster0.ext_abort_fill_data	int	-0x202020303020203	Returned data, if external aborts are asynchronous
subcluster0.ext_abort_normal_cacheable_read_is_critical	bool	0x0	Critical reporting of normal write-back cacheable-read external aborts
subcluster0.ext_abort_normal_cacheable_read_is_sync	bool	0x1	Synchronous reporting of normal write-back cacheable-read external aborts
subcluster0.ext_abort_normal_cacheable_read_ras_index	int	0x0	External Aborts are reported in RAS record index specified in this param. Values: Valid indices in range [0, number_of_error_records).
subcluster0.ext_abort_normal_cacheable_read_ras_type	int	0x0	External Aborts are reported as RAS error type specified in this param. Values: 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
subcluster0.ext_abort_normal_cacheable_write_is_critical	bool	0x0	Critical reporting of normal write-back cacheable write external aborts
subcluster0.ext_abort_normal_cacheable_write_is_sync	bool	0x0	Synchronous reporting of normal write-back cacheable write external aborts
subcluster0.ext_abort_normal_cacheable_write_ras_index	int	0x0	External Aborts are reported in RAS record index specified in this param. Values: Valid indices in range [0, number_of_error_records).
subcluster0.ext_abort_normal_cacheable_write_ras_type	int	0x0	External Aborts are reported as RAS error type specified in this param. Values: 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
subcluster0.ext_abort_normal_noncacheable_prefetch_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_index, Valid indices in range [0, number_of_error_records).
subcluster0.ext_abort_normal_noncacheable_prefetch_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-80 Parameters for Subcluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
subcluster0.ext_abort_normal_noncacheable_read_is_critical	bool	0x0	Critical reporting of normal noncacheable-read external aborts
subcluster0.ext_abort_normal_noncacheable_read_is_sync	bool	0x1	Synchronous reporting of normal noncacheable-read external aborts
subcluster0.ext_abort_normal_noncacheable_read_ras_index	int	0x0	External Aborts are reported in RAS record index specified in this param. Values: Valid indices in range [0, number_of_error_records).
subcluster0.ext_abort_normal_noncacheable_read_ras_type	int	0x0	External Aborts are reported as RAS error type specified in this param. Values: 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
subcluster0.ext_abort_normal_noncacheable_write_is_critical	bool	0x0	Critical reporting of normal noncacheable write external aborts
subcluster0.ext_abort_normal_noncacheable_write_is_sync	bool	0x0	Synchronous reporting of normal noncacheable write external aborts
subcluster0.ext_abort_normal_noncacheable_write_ras_index	int	0x0	External Aborts are reported in RAS record index specified in this param. Values: Valid indices in range [0, number_of_error_records).
subcluster0.ext_abort_normal_noncacheable_write_ras_type	int	0x0	External Aborts are reported as RAS error type specified in this param. Values: 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
subcluster0.ext_abort_normal_wt_cacheable_prefetch_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_index, Valid indices in range [0, number_of_error_records).
subcluster0.ext_abort_normal_wt_cacheable_prefetch_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
subcluster0.ext_abort_normal_wt_cacheable_read_is_critical	bool	0x0	Critical reporting of normal write-through cacheable-read external aborts
subcluster0.ext_abort_normal_wt_cacheable_read_is_sync	int	0x2	Synchronous reporting of normal write-through read external aborts. 0, asynchronous. 1, synchronous. 2, same as ext_abort_normal_cacheable_read_is_sync.
subcluster0.ext_abort_normal_wt_cacheable_read_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_normal_cacheable_read_ras_index, Valid indices in range [0, number_of_error_records).
subcluster0.ext_abort_normal_wt_cacheable_read_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_normal_cacheable_read_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
subcluster0.ext_abort_normal_wt_cacheable_write_is_critical	bool	0x0	Critical reporting of normal write-through write external aborts

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-80 Parameters for Subcluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
subcluster0.ext_abort_normal_wt_cacheable_write_is_sync	int	0x2	Synchronous reporting of normal write-through write external aborts. 0, asynchronous. 1, synchronous. 2, same as ext_abort_normal_cacheable_write_is_sync.
subcluster0.ext_abort_normal_wt_cacheable_write_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_normal_cacheable_write_ras_index, Valid indices in range [0, number_of_error_records).
subcluster0.ext_abort_normal_wt_cacheable_write_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_normal_cacheable_write_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
subcluster0.ext_abort_prefetch_device_GRE_read_is_critical	bool	0x0	Critical reporting of external aborts generated by device-GRE instruction fetches
subcluster0.ext_abort_prefetch_device_GRE_read_is_sync	int	0x2	Behaviour of external aborts generated by device-GRE instruction fetches. 0, asynchronous abort. 1, synchronous abort. 2, same as ext_abort_prefetch_is_sync.
subcluster0.ext_abort_prefetch_device_nGRE_read_is_critical	bool	0x0	Critical reporting of external aborts generated by device-nGRE instruction fetches
subcluster0.ext_abort_prefetch_device_nGRE_read_is_sync	int	0x2	Behaviour of external aborts generated by device-nGRE instruction fetches. 0, asynchronous abort. 1, synchronous abort. 2, same as ext_abort_prefetch_is_sync.
subcluster0.ext_abort_prefetch_device_read_is_critical	bool	0x0	Critical reporting of external aborts generated by device-nGnRE instruction fetches
subcluster0.ext_abort_prefetch_device_read_is_sync	int	0x2	Behaviour of external aborts generated by device-nGnRE instruction fetches. 0, asynchronous abort. 1, synchronous abort. 2, same as ext_abort_prefetch_is_sync.
subcluster0.ext_abort_prefetch_is_critical	bool	0x0	Critical reporting of external aborts generated by normal writeback cacheable instruction fetches
subcluster0.ext_abort_prefetch_is_sync	bool	0x1	Behaviour of external aborts generated by normal writeback cacheable instruction fetches. 0, asynchronous abort. 1, synchronous abort.
subcluster0.ext_abort_prefetch_noncacheable_read_is_critical	bool	0x0	Critical reporting of external aborts generated by normal noncacheable instruction fetches
subcluster0.ext_abort_prefetch_noncacheable_read_is_sync	int	0x2	Behaviour of external aborts generated by normal noncacheable instruction fetches. 0, asynchronous abort. 1, synchronous abort. 2, same as ext_abort_prefetch_is_sync.
subcluster0.ext_abort_prefetch_ras_index	int	0x0	External Aborts are reported in RAS record index specified in this param. Values: Valid indices in range [0, number_of_error_records).
subcluster0.ext_abort_prefetch_ras_type	int	0x0	External Aborts are reported as RAS error type specified in this param. Values: 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-80 Parameters for Subcluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
subcluster0.ext_abort_prefetch_so_read_is_critical	bool	0x0	Critical reporting of external aborts generated by device-nGnRnE instruction fetches
subcluster0.ext_abort_prefetch_so_read_is_sync	int	0x2	Behaviour of external aborts generated by device=nGnRnE instruction fetches. 0, asynchronous abort. 1, synchronous abort. 2, same as ext_abort_prefetch_is_sync.
subcluster0.ext_abort_prefetch_wt_cacheable_read_is_critical	bool	0x0	Critical reporting of external aborts generated by normal writethrough cacheable instruction fetches
subcluster0.ext_abort_prefetch_wt_cacheable_read_is_sync	int	0x2	Behaviour of external aborts generated by normal writethrough cacheable instruction fetches. 0, asynchronous abort. 1, synchronous abort. 2, same as ext_abort_prefetch_is_sync.
subcluster0.ext_abort_so_prefetch_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_index, Valid indices in range [0, number_of_error_records).
subcluster0.ext_abort_so_prefetch_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
subcluster0.ext_abort_so_read_is_critical	bool	0x0	Critical reporting of device-nGnRnE read external aborts
subcluster0.ext_abort_so_read_is_sync	bool	0x1	Synchronous reporting of device-nGnRnE read external aborts
subcluster0.ext_abort_so_read_ras_index	int	0x0	External Aborts are reported in RAS record index specified in this param. Values: Valid indices in range [0, number_of_error_records).
subcluster0.ext_abort_so_read_ras_type	int	0x0	External Aborts are reported as RAS error type specified in this param. Values: 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
subcluster0.ext_abort_so_write_is_critical	bool	0x0	Critical reporting of device-nGnRnE write external aborts
subcluster0.ext_abort_so_write_is_sync	bool	0x1	Synchronous reporting of device-nGnRnE write external aborts
subcluster0.ext_abort_so_write_ras_index	int	0x0	External Aborts are reported in RAS record index specified in this param. Values: Valid indices in range [0, number_of_error_records).
subcluster0.ext_abort_so_write_ras_type	int	0x0	External Aborts are reported as RAS error type specified in this param. Values: 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
subcluster0.ext_abort_ttw_cacheable_read_is_critical	bool	0x0	Critical reporting of TTW cacheable read external aborts
subcluster0.ext_abort_ttw_cacheable_read_is_sync	bool	0x1	Synchronous reporting of TTW cacheable read external aborts
subcluster0.ext_abort_ttw_cacheable_read_ras_index	int	0x0	External Aborts are reported in RAS record index specified in this param. Values: Valid indices in range [0, number_of_error_records).

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-80 Parameters for Subcluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
subcluster0.ext_abort_ttw_cacheable_read_ras_type	int	0x0	External Aborts are reported as RAS error type specified in this param. Values: 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
subcluster0.ext_abort_ttw_noncacheable_read_is_critical	bool	0x0	Critical reporting of TTW noncacheable read external aborts
subcluster0.ext_abort_ttw_noncacheable_read_is_sync	bool	0x1	Synchronous reporting of TTW noncacheable read external aborts
subcluster0.ext_abort_ttw_noncacheable_read_ras_index	int	0x0	External Aborts are reported in RAS record index specified in this param. Values: Valid indices in range [0, number_of_error_records).
subcluster0.ext_abort_ttw_noncacheable_read_ras_type	int	0x0	External Aborts are reported as RAS error type specified in this param. Values: 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
subcluster0.ext_abort_ttw_wt_cacheable_read_is_critical	bool	0x0	Critical reporting of TTW write-through cacheable read external aborts
subcluster0.ext_abort_ttw_wt_cacheable_read_is_sync	int	0x2	Synchronous reporting of TTW write-through cacheable read external aborts. 0, asynchronous. 1, synchronous. 2, same as ext_abort_ttw_cacheable_read_is_sync.
subcluster0.ext_abort_ttw_wt_cacheable_read_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_ttw_cacheable_read_ras_index, Valid indices in range [0, number_of_error_records).
subcluster0.ext_abort_ttw_wt_cacheable_read_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_ttw_cacheable_read_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
subcluster0.fault_on_nT_bit_set	bool	0x1	Whether block translation table entries with the nT bit set should always fault. Only applies when changing_block_size_without_bbm_support_level is 1 or higher.
subcluster0.fault_unalign_to_unsupported_mem	int	0x0	If has_unaligned_single_copy_atomicity is true, whether unaligned A64 atomic, exclusive and acquire/release instructions to non iWB-oWB generate fault. This parameter is a bitmap. Bits 0,1,2 should be set accordingly to enable the fault behaviour. bit 0(LSB): atomic access should fault, bit 1: exclusive access should fault, bit 2: acquire/release should fault.
subcluster0.fault_unaligned_s1_device_s2_fwb	bool	0x0	Whether unaligned access with stage1 Device memory and final memory attribute forced to normal by FWB can generate alignment fault
subcluster0.force_deterministic_irg_tag_generation	bool	0x0	Force the random tag generated by the IRG instruction when GCR_EL1.RRND=1 to equal RGSR_EL1.SEED[3:0] rather than a non-deterministic value
subcluster0.fpcr_short_vector_raz	bool	0x0	FPSCR and FPCR fields LEN and STRIDE are hardwired to 0

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-80 Parameters for Subcluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
subcluster0.fsr_ext_bit_update_kind	int	0x3	Set/Clear DFSR/IFSR EA bit on Synchronous/Async External Aborts. 0: Never Set, 1: Set on Synchronous Ext Aborts 2: Set on Asynchronous Ext Aborts 3: Set on both Sync and Async Ext Aborts.
subcluster0.global_debug_rom.ROMDEVID	int	0x0	Value of Debug Rom Device Identification Register
subcluster0.global_debug_rom.ROMPIDR	int	0x4000bb000	Value of Debug Rom Peripheral Identification Register
subcluster0.hardware_translation_table_update_implemented	int	0x1	Implement hardware translation table updates from ARMv8.1. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.1 is enabled. - 2, feature is implemented.
subcluster0.has_16bit_asids	bool	0x1	Enable 16-bit ASIDs.
subcluster0.has_16bit_vmids	int	0x1	Implement support for 16-bit VMIDs from ARMv8.1. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.1 is enabled. - 2, feature is implemented.
subcluster0.has_16k_granule	bool	0x0	Implement the 16k LPAE translation granule.
subcluster0.has_4k_granule	bool	0x1	Implement the 4k LPAE translation granule.
subcluster0.has_52bit_address_with_16k	int	0x0	Implements Armv8.7 52-bit IPA/PA support for 16k Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if Armv8.7 is enabled. - 2, feature is implemented.
subcluster0.has_52bit_address_with_4k	int	0x0	Implements Armv8.7 52-bit IPA/PA support for 4k Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if Armv8.7 is enabled. - 2, feature is implemented.
subcluster0.has_64k_granule	bool	0x1	Implement the 64k LPAE translation granule.
subcluster0.has_aarch32_hpd	bool	0x0	If true then hierarchical permission disable is supported in AArch32
subcluster0.has_aarch64	bool	0x1	All implemented exception levels can run in AArch64
subcluster0.has_actlr2	bool	0x0	If true ACLTR2 exists and ACTLR2(NS) is aliased to ACTLR_EL1[63:32]
subcluster0.has_amu	int	0x0	Implement activity monitor functionality from ARMv8.4. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
subcluster0.has_arm_v8-1	bool	0x0	Implement the ARMv8.1 Extension.
subcluster0.has_arm_v8-2	bool	0x0	Implement the ARMv8.2 Extension.
subcluster0.has_arm_v8-3	bool	0x0	Implement the ARMv8.3 Extension.
subcluster0.has_arm_v8-4	bool	0x0	Implement the ARMv8.4 Extension.
subcluster0.has_arm_v8-5	bool	0x0	Implement the ARMv8.5 Extension.
subcluster0.has_arm_v8-6	bool	0x0	Implement the ARMv8.6 Extension.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-80 Parameters for Subcluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
subcluster0.has_arm_v8-7	bool	0x0	Implement the Armv8.7 Extension. This feature is incomplete and under development.
subcluster0.has_at_with_pan	int	0x1	Implement new AT instructions with PAN support. Possible values of this parameter are: - 1, feature is implemented if ARMv8.2 is enabled. - 2, feature is implemented.
subcluster0.has_axflag_xaflag	int	0x1	Implement flag manipulation instructions (AXFlag, XAFlag) from ARMv8.5. Possible values of this parameter are: - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
subcluster0.has_axflag_xaflag_frint	int	0x1	Implement flag manipulation instructions (AXFlag, XAFlag) and floating-point rounding to int instructions (FRINT[32|64][X|Z]) from ARMv8.5. If this parameter is enabled, it also enables both has_axflag_xaflag and has_frint. If support for only one of the features is needed, please use the individual parameters and do not enable this one. Possible values of this parameter are: - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
subcluster0.has_branch_target_exception	int	0x0	Implement Branch target identification mechanism from ARMv8.5. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
subcluster0.has_ccidx	bool	0x0	Implement the ARMv8.3 CCSIDR Extension. Extending the ccsidr number of sets.
subcluster0.has_cfinv_rmif_setf	int	0x1	Implement flag manipulation (CFINV, RMIF, SETF8, SETF16) instructions from ARMv8.4. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
subcluster0.has_coherent_icache	bool	0x0	Whether icache invalidation to the point of unification is required for instruction to data coherence. true - Invalidate operations not required
subcluster0.has_common_not_private_translations	int	0x1	Implement the TTBRn_ELx.CnP (Common not Private) controls from ARMv8.2. Possible values of this parameter are: - 1, feature is implemented if ARMv8.2 is enabled. - 2, feature is implemented.
subcluster0.has_complex_number	int	0x1	Implement ARMv8.3 complex number support, Multiply Accumulate and Add instructions. Possible values of this parameter are: - 1, feature is implemented if ARMv8.3 is enabled. - 2, feature is implemented.
subcluster0.has_cvadp_support	int	0x0	Implement instruction to support cache clean by deep persistence (DC CVADP) from ARMv8.5, can be selected for core implemented on any arch version starting ARMv8.2 Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.2 is enabled. - 2, feature is implemented.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-80 Parameters for Subcluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
subcluster0.has_data_alignment_flag	int	0x0	Implement non-optimal misalignment flag for PMU/SPE from ARMv8.5 Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
subcluster0.has_delayed_ctireg	bool	0x0	Delay the functional effect of CTI register writes until ISB or implicit barrier.
subcluster0.has_delayed_dbgreg	bool	0x0	Delay the functional effect of external debug register writes until ISB or implicit barrier.
subcluster0.has_delayed_sysreg	bool	0x0	Delay the functional effect of system register writes until ISB or implicit barrier.
subcluster0.has_delayed_wfe_trap	int	0x0	Implements Configurable Delayed WFE trapping from ARMv8.6 Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.6 is enabled. - 2, feature is implemented.
subcluster0.has_dgh	int	0x1	Implements Data Gathering Hint instruction from ARMv8.6 Possible values of this parameter are: - 1, feature is implemented if ARMv8.6 is enabled. - 2, feature is implemented.
subcluster0.has_dot_product	int	0x1	Implement the dot product (UDOT, SDOT) instructions from ARMv8.4. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
subcluster0.has_e0pd	int	0x1	Implement ARMv8.5 feature to prevent unprivileged access to one half of the memory Possible values of this parameter are: - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
subcluster0.has_el2	bool	0x1	Implements EL2
subcluster0.has_el3	bool	0x1	Implements EL3
subcluster0.has_enhanced_pac	bool	0x0	If pointer authentication is enabled then implement enhanced PAC.
subcluster0.has_enhanced_pan	int	0x1	Implements Armv8.7 Enhanced PAN feature Possible values of this parameter are: - 1, feature is implemented if Armv8.7 is enabled. - 2, feature is implemented.
subcluster0.has_ets	bool	0x0	Whether Enhanced Translation Synchronization is supported
subcluster0.has_extended_recp_estimate	int	0x0	Implements increased precision of reciprocal instructions.) Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if Armv8.7 is enabled. - 2, feature is implemented.
subcluster0.has_fgt	int	0x1	Implements Fine-grained Virtualization Traps extension from ARMv8.6 Possible values of this parameter are: - 1, feature is implemented if ARMv8.6 is enabled. - 2, feature is implemented.
subcluster0.has_fp16	int	0x1	Implement the half-precision floating-point data processing instructions from ARMv8.2. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.2 is enabled. - 2, feature is implemented.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-80 Parameters for Subcluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
subcluster0.has_fp16_fmlal	int	0x1	Implement the New Floating Point Multiplication Variant (FP16 FMLAL, FMLSL) instructions from ARMv8.4. Only supported if has_fp16=0x1. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
subcluster0.has_frint	int	0x1	Implement floating-point rounding to int instructions (FRINT[32|64][X|Z]) from ARMv8.5. Possible values of this parameter are: - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
subcluster0.has_generic_authentication	int	0x1	Implement ARMv8.3 generic authentication. Possible values of this parameter are: - 1, feature is implemented if ARMv8.3 is enabled. - 2, feature is implemented.
subcluster0.has_guest_translation_granule	int	0x1	Implement mechanism for guest translation granule identification from ARMv8.5, ID values determined by stage1 granule configuration parameters Possible values of this parameter are: - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
subcluster0.has_hardware_translation_table_update	int	0x2	Type of hardware translation table supported (when enabled by hardware_translation_table_update_implemented). 0, not implemented. 1, access bit updates implemented. 2, access bit updates and dirty bit mechanism implemented.
subcluster0.has_hcrx_el2	int	0x1	Implements new HCRX_EL2 id register from Armv8.7 Possible values of this parameter are: - 1, feature is implemented if Armv8.7 is enabled. - 2, feature is implemented.
subcluster0.has_id_reg_read	int	0x1	Implement read access to the ID registers (ESR_ELx.EC=0x18) Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
subcluster0.has_iesb	int	0x1	Implement support for implicit error sync event from ARMv8.2. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.2 is enabled. - 2, feature is implemented.
subcluster0.has_jscvt	int	0x1	Implement ARMv8.3 javascript Floating-point to Integer conversion instruction. Possible values of this parameter are: - 1, feature is implemented if ARMv8.3 is enabled. - 2, feature is implemented.
subcluster0.has_large_system_ext	bool	0x0	Implement the ARMv8 Large System Extensions.
subcluster0.has_large_va	int	0x0	Implement support for the extended 52-bit virtual addresses from ARMv8.2. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.2 is enabled. - 2, feature is implemented.
subcluster0.has_ldapur_stlur	int	0x1	Implement support for LDAPR and STLR instructions with immediate offsets from ARMv8.4. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-80 Parameters for Subcluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
subcluster0.has_ldm_stm_ordering_control	int	0x0	Implement the SCTLR_ELx.LSMAOE (Load/Store Multiple Atomicity and Ordering Enable) and SCTLR_ELx,nTLSMD (no Trap Load/Store Multiple to Device) controls from ARMv8.2. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.2 is enabled. - 2, feature is implemented.
subcluster0.has_lrcpc	bool	0x0	If true then it support the RCpc feature (ARMv8.3)
subcluster0.has_mpam	int	0x0	Implement ARMv8.4 MPAM Registers and associated functionality. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
subcluster0.has_mt_pmu_disable_feature	int	0x0	Implements Multi-threading PMU disable extension from ARMv8.6 Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.6 is enabled. - 2, feature is implemented.
subcluster0.has_mte_ctrl_bits_stateful	bool	0x0	if memory_tagging_support_level == 1, Whether mte specific control bits in system registers are stateful
subcluster0.has_nested_virtualization	int	0x1	Implement ARMv8.3 nested virtualization. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.3 is enabled. - 2, feature is implemented.
subcluster0.has_no_os_double_lock	int	0x0	Do not implement the OS double-lock. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
subcluster0.has_non_context_synchronizing_exception_controls	int	0x1	Implement cosmetic controls for whether exception entry and exit are context synchronizing events (SCTLR_ELx.{EIS,EOS}) from ARMv8.5. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
subcluster0.has_par_bit10_razwi	bool	0x0	Whether PAR_EL1[10] is RAZ/WI
subcluster0.has_pointer_authentication	int	0x1	Implement ARMv8.3 pointer authentication. Possible values of this parameter are: - 1, feature is implemented if ARMv8.3 is enabled. - 2, feature is implemented.
subcluster0.has_prediction_invalidation_instructions	int	0x1	Implement execution and data prediction invalidation from ARMv8.5 Possible values of this parameter are: - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
subcluster0.has_pstate_dit	int	0x1	Implement timing insensitivity of data processing instructions from ARMv8.4. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
subcluster0.has_pstate_pan	int	0x1	Implement the PSTATE.PAN (Privileged Access Never) control from ARMv8.1 Possible values of this parameter are: - 1, feature is implemented if ARMv8.1 is enabled. - 2, feature is implemented.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-80 Parameters for Subcluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
subcluster0.has_pstate_uao	int	0x1	Implement the PSTATE.UAO (User Access Override) control from ARMv8.2. Possible values of this parameter are: - 1, feature is implemented if ARMv8.2 is enabled. - 2, feature is implemented.
subcluster0.has_ras_armv84_extension	int	0x1	Implement ARMv8.4 RAS Extension. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
subcluster0.has_ras_critical_error	int	0x0	[DEPRECATED: Set CI field on first register in error_record_feature_register JSON instead] ARMv8.4 AArch64 RAS Critical Error is implemented or not. 0 - Feature Not Supported, 1 - Feature always enabled, 2 - Feature is controllable.
subcluster0.has_ras_double_fault	int	0x1	Implement ARMv8.4 RAS Doublt Fault Extension. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
subcluster0.has_ras_fault_injection	int	0x0	[DEPRECATED: Set INJ field on first register in error_record_feature_register JSON instead] Implement ARMv8.4 Standard Fault Injection mechanism. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
subcluster0.has_ras_timestamp	int	0x0	[DEPRECATED: Set TS field on first register in error_record_feature_register JSON instead] ARMv8.4 AArch64 RAS Timestamp register is implemented or not. 0 - No Timestamp is recorded, 1 - Generic Timer timestamp is recorded, 2 - IMP DEF timestamp is recorded.
subcluster0.has_restriction_on_speculative_data_loaded	int	0x1	Implements the ARMv8.5 security feature (Restrictions on the effects of speculation) Possible values of this parameter are: - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
subcluster0.has_rndr	int	0x0	Implement random number instructions to read from RNDR and RNDRSS random number registers from ARMv8.5. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
subcluster0.has_rounding_doubling_multiply_add_subtract	int	0x1	Implement the rounding doubling multiply add and subtract instructions from ARMv8.1 Possible values of this parameter are: - 1, feature is implemented if ARMv8.1 is enabled. - 2, feature is implemented.
subcluster0.has_secure_el2	int	0x1	Implement support for Secure EL2 Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
subcluster0.has_self_hosted_trace_extension	int	0x1	Implement support for the Self-hosted Trace Extensions from ARMv8.4. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-80 Parameters for Subcluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
subcluster0.has_small_page_table	int	0x1	Implement small page table support which increases the maximum value of TxSZ field from ARMv8.4. Note: will be unimplemented only if both has_small_page_table=0x0 and has_secure_el2=0x0. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
subcluster0.has_speculation_barrier_inst	int	0x1	Implement speculation barrier instruction (SB) from ARMv8.5 Possible values of this parameter are: - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
subcluster0.has_stage2_fwb	int	0x1	Implement HCR_EL2.FWB, stage 2 control of memory types and cacheability Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
subcluster0.has_stage2_xnx	int	0x1	Implement the extended XN[1:0] stage 2 control from ARMv8.2. Possible values of this parameter are: - 1, feature is implemented if ARMv8.2 is enabled. - 2, feature is implemented.
subcluster0.has_stage2_xnx_in_aarch32	bool	0x1	Implement the extended XN[1:0] stage 2 control from ARMv8.2 for Aarch32.
subcluster0.has_statistical_profiling	bool	0x1	Whether Statistical Based Profiling is implemented
subcluster0.has_supersections	bool	0x1	Whether VMSAv8-32 supersection to support more than 32-bit PA using short descriptor is implemented.
subcluster0.has_synchronous_load_atomics	bool	0x1	Report asynchronous abort due to unsupported load atomics as synchronous
subcluster0.has_synchronous_store_atomics	bool	0x0	Report asynchronous abort due to unsupported store atomics as synchronous
subcluster0.has_tlb_conflict_abort	bool	0x0	Detected inconsistent TLB content generate aborts.
subcluster0.has_tlbi_range	int	0x1	Implement support for TLB Range Maintenance instructions (TLBI RVAE1, etc) from ARMv8.4. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
subcluster0.has_tlbi_to_outer_shareable	int	0x1	Implement support for TLB Maintenance instructions that extend to the Outer Shareable domain (TLBI VAE1OS, etc) from ARMv8.4. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
subcluster0.has_tlbi_ttl	int	0x1	Implement support for the TTL level hint in by-address TLB Maintenance instructions from ARMv8.4. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-80 Parameters for Subcluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
subcluster0.has_unaligned_single_copy_atomicity	int	0x1	Implement support for SCTLR_ELx.nAA from ARMv8.4, and A64 atomic, exclusive and acquire/release instructions accessing unaligned bytes inside a 16byte window will not generate alignment fault. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
subcluster0.has_unsupported_exclusive_fault	bool	0x1	Report unsupported exclusive access with Unsupported Exclusive fault status (otherwise use external abort)
subcluster0.has_v8_4_debug_extension	int	0x1	Implement ARMv8.4 debug extensions Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
subcluster0.has_v8_4_pmu_extension	int	0x1	Implement PMU extension from ARMv8.4. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
subcluster0.has_v8_5_debug_over_power_down	int	0x0	Implement ARMv8.5 Debug over powerdown Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
subcluster0.has_v8_5_pmu_extension	int	0x1	Implement PMU extension from ARMv8.5. Possible values of this parameter are: - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
subcluster0.has_v8_5_spe_extension	int	0x1	Implement SPE extension from ARMv8.5 Possible values of this parameter are: - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
subcluster0.has_v8_6_pmu_events	int	0x1	Implements PMU events from ARMv8.6 Possible values of this parameter are: - 1, feature is implemented if ARMv8.6 is enabled. - 2, feature is implemented.
subcluster0.has_v8_7_fp_enhancements	int	0x1	Implements the Floating Point enhancements from Armv8.7 (introduces FPCR.FIZ/AH/NEP, etc.) Possible values of this parameter are: - 1, feature is implemented if Armv8.7 is enabled. - 2, feature is implemented.
subcluster0.has_v8_7_pmu_events	int	0x1	Implement PMU events from ARMv8.7. Possible values of this parameter are: - 1, feature is implemented if Armv8.7 is enabled. - 2, feature is implemented.
subcluster0.has_v8_7_pmu_extension	int	0x1	Implement PMU extension from ARMv8.7. Possible values of this parameter are: - 1, feature is implemented if Armv8.7 is enabled. - 2, feature is implemented.
subcluster0.has_v8_7_spe_extension	int	0x1	Implement SPE extension from ARMv8.7. Possible values of this parameter are: - 1, feature is implemented if Armv8.7 is enabled. - 2, feature is implemented.
subcluster0.has_vncr_el2	int	0x1	Implement support for nested virtualization enhancements from ARMv8.4. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-80 Parameters for Subcluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
subcluster0.has_wfet_and_wfit	int	0x1	Implements WFE and WFI with Timeout from Armv8.7 Possible values of this parameter are: - 1, feature is implemented if Armv8.7 is enabled. - 2, feature is implemented.
subcluster0.has_writebuffer	bool	0x0	Implement write accesses buffering before L1 cache. May affect ext_abort behaviour.
subcluster0.has_xs	int	0x1	Implements Armv8.7 XS, TLBInXS, DSBnXS instruction Possible values of this parameter are: - 1, feature is implemented if Armv8.7 is enabled. - 2, feature is implemented.
subcluster0.hsr_uncond_cc	bool	0x0	Condition codes reported in HSR as AL if it passes
subcluster0.icache-hit_latency	int	0x0	L1 I-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when icache-state_modelled=true.
subcluster0.icache-log2linelen	int	0x0	If nonzero, Log2 of the instruction cache line length in bytes (valid values in range 4-8). Otherwise the value of cache-log2linelen is used.
subcluster0.icache-maintenance_latency	int	0x0	L1 I-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when icache-state_modelled=true.
subcluster0.icache-miss_latency	int	0x0	L1 I-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when icache-state_modelled=true.
subcluster0.icache-nprefetch	int	0x1	Number of next sequential instruction cache lines to prefetch. This is only used when icache-prefetch_enabled=true.
subcluster0.icache-prefetch_enabled	bool	0x0	Enable simulation of instruction cache prefetching. This is only used when icache-state_modelled=true.
subcluster0.icache-prefetch_level	int	0x0	0 based cache level at which instructions are pre-fetched. This is only used when icache-prefetch_enabled=true.
subcluster0.icache-read_access_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per access (of size icache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if icache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when icache-state_modelled=true.
subcluster0.icache-read_bus_width_in_bytes	int	0x8	L1 I-Cache read bus width in bytes used to calculate per-access timing annotations
subcluster0.icache-read_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per byte accessed.icache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when icache-state_modelled=true.
subcluster0.icache-size	int	0x8000	L1 I-Cache size in bytes.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-80 Parameters for Subcluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
subcluster0.icache-ways	int	0x2	L1 I-Cache number of ways (sets are implicit from size).
subcluster0.ignore_DBGPRCR_CWRR	bool	0x0	Ignore writes to the deprecated DBGPRCR.CWRR bit.
subcluster0.ignore_access_flag_update_by_at_ops	bool	0x0	If true, AT operations do not update access flag
subcluster0.ignore_large_address_top_bits_in_page_walk	bool	0x0	Whether page table bits [15:12] are ignored if PA_SIZE < 52 and output address is configured < 52 with large page
subcluster0.imp_def_functionality_behaviour	int	0x0	Behaviour of IMPLEMENTATION DEFINED registers and system instructions. 0, UNDEF. 1, RAZ/WI.
subcluster0.impdef_regs_and_unpred_from_implementation	string	""	Configure implementation defined registers and unpredictable behaviour to match the specified implementation. Requires a license for the selected implementation model. Use ARM_Cortex-A<num> or ARM_<codename> for licensed pre-release cores.
subcluster0.independent_cache_control_traps	int	0x0	Implement Independent Cache Control traps from ARMv8.5. 0, NO_SUPPORT. 1, SUPPORTED_BUT_NOT_FOR_TLB_MAINTENANCE_INSTRUCTIONS. 2, FULL_SUPPORT.
subcluster0.insert_iesb_before_exception	bool	0x0	If true then inserts an IESB before taking with Exception otherwise has no effect and IESB is taken after PState is changed due to the Exception.
subcluster0.instruction_tlb_size	int	0x0	Number of stage1+2 itlb entries (or 0 for unified ITLB+DTLB)
subcluster0.is_mt_res0	bool	0x0	If ARMv8.6 is not implemented, and PMUv3 is implemented, this parameter controls whether PMEVTYPER<n>.MT bit is RES0 or RW. For other implementations, this parameter has no effect
subcluster0.is_uniprocessor	bool	0x0	Value for the U bit in MPIDR. true disables L1 cache coherency protocols
subcluster0.max_32bit_el	int	0x3	Maximum exception level supporting AArch32 modes.
subcluster0.memory.acp.AxCACHE_mask	int	0x0	Used with memory.acp.AxCACHE_pattern to define which memory types the ACP port accepts. All transactions which do not satisfy (AxCACHE & mask) == pattern will abort.
subcluster0.memory.acp.AxCACHE_pattern	int	0x0	Used with memory.acp.AxCACHE_mask to define which memory types the ACP port accepts. All transactions which do not satisfy (AxCACHE & mask) == pattern will abort.
subcluster0.memory.l2_cache.is_inner_cacheable	bool	0x1	L2 cache obeys inner cacheable attributes (rather than outer cacheable attributes)
subcluster0.memory.l2_cache.is_inner_shareable	bool	0x1	L2 cache obeys inner shareable attributes (rather than outer sharaable attributes)
subcluster0.memory_tagging_support_level	int	0x0	Specify the memory tagging extension support level: 0, not implemented. 1, instructions and registers only are implemented. 2, implemented. 3, implemented with asymmetric handling of exceptions

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-80 Parameters for Subcluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
subcluster0.mixed_endian	int	0x1	Implement support for mixed endianness. 0, not supported. 1, supported at all exception levels. 2, supported at EL0 only.
subcluster0.mpam_force_ns_rao	bool	0x0	Whether MPAM3_EL3.FORCE_NS bit is RAO/WI.
subcluster0.mpam_frac	int	0x0	MPAM fractional revision number in ID_AA64PFR1_EL1.MPAM_frac field
subcluster0.mpam_has_hcr	bool	0x0	MPAM Whether MPAMIDR_EL1 HAS_HCR bit is set or clear
subcluster0.mpam_max_partid	int	0xffff	MPAM Maximum PARTID Supported
subcluster0.mpam_max_pmg	int	0xff	MPAM Maximum PMG Supported
subcluster0.mpam_max_vpmr	int	0x0	MPAM Maximum VPMR Supported
subcluster0.mpamidr_has_force_ns	int	0x0	Whether MPAMIDR_EL1.HAS_FORCE_NS bit is set or clear. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.6 is enabled. - 2, feature is implemented.
subcluster0.mpamidr_has_sdeflt	int	0x0	Whether MPAMIDR_EL1.HAS_SDEFLT bit is set or clear. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.6 is enabled. - 2, feature is implemented.
subcluster0.mpamidr_has_tidr	int	0x0	Whether MPAMIDR_EL1.HAS_TIDR bit is set or clear Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.6 is enabled. - 2, feature is implemented.
subcluster0.mvbar_reset_is_rvbar	bool	0x1	If true then the reset value of MVBAR is RVBAR, if false the reset value is UNKNOWN.
subcluster0.non_secure_vgic_alias_when_ns_only	int	0x0	If ! has_el3 and only non-secure side exists, then the normal position of the VGIC is a secure alias. If this parameter is non-zero then in addition a non-secure alias of the VGIC will be placed at this position (aligned to 32 KB).
subcluster0.num_loregion_descriptors	int	0x0	Number of Limited Ordering Region descriptors implemented (if ARM v8.1 extensions are implemented)
subcluster0.num_loregions	int	0x0	Number of Limited Ordering Regions implemented excluding background region (if ARM v8.1 extensions are implemented)
subcluster0.number_of_error_records	int	0x0	Cores Number of Error records supported for RAS
subcluster0.optimal_alignment_size	int	0x1	Alignment boundary which does not incur additional performance penalty from ARMv8.5. - 1, architectural misalignment is used to set PMU event LDST_ALIGN_LAT and SPE event E[11] - 2, access crossing 4 byte boundary is used to set PMU event LDST_ALIGN_LAT and SPE event E[11] - 3, access crossing 8 byte boundary is used to set PMU event LDST_ALIGN_LAT and SPE event E[11] ... - 12, access crossing 4 Kbyte boundary is used to set PMU event LDST_ALIGN_LAT and SPE event E[11]

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-80 Parameters for Subcluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
subcluster0.output_attributes	string	"ExtendedID[62:55]=MPAM_PMG, ExtendedID[54:39]=MPAM_PARTID, ExtendedID[38]=MPAM_NS"	User-defined transform to be applied to bus attributes like MasterID, ExtendedID or UserFlags. Currently, only works for MPAM Attributes encoding into bus attributes.
subcluster0.page_based_hardware_attributes	int	0x0	Implement the page based hardware attributes from ARMv8.2. This parameter indicates which page table bits are available for hardware, where bits[3:0] correspond to PTE[62:59] and to TCR_ELx.HWUnyy.
subcluster0.pmmir_el1_bus_slots	int	0x0	Largest value by which BUS_ACCESS can increment over BUS_CYCLES cycles. From v8.7 PMU extension
subcluster0.pmmir_el1_bus_width	int	0x0	Width, in bytes, of accesses counted by BUS_ACCESS. From v8.7 PMU extension
subcluster0.pmu-num_counters	int	0x8	Number of PMU counters implemented
subcluster0.pseudo_fault_generation_feature_register	string	""	ARMv8.4 Standard Pseudo-fault generation feature register values. JSON schema for the parameter value is: [{"OF":false,"UC":false,"UEU":false,"UER":false,"UEO":false,"DE":false,"CE":0x0,"CI":false,"ER":false,"PN":false,"AV":false,"MV":false,"SYN":false,"R":false},other_psuedo-fault_generating_features_register_values]. Where OF, UC, UEU, UER, UEO, DE, CI, ER, PN, AV, MV, SYN, and R have valid false(NOT_SUPPORTED) and true(FEATURE_CONTROLLABLE), where CE can have 0(NOT_SUPPORTED), 1(NONSPECIFIC_CE_SUPPORTED) and 3(TRANSIENT_OR_PERSISTENT_CE_SUPPORTED). Effective only when ERXFR's INJ field allows it or has_ras_fault_injection is true.
subcluster0.pstate_ssbs_type	int	0x0	Implement speculative store bypass safe feature from ARMv8.5. 0, Not supported. 1, Supported without MSR/MRS access to SSBS. 2, fully supported.
subcluster0.ptw_latency	int	0x0	Page table walker latency for TA (Timing Annotation), expressed in simulation ticks
subcluster0.randomize_unknowns_at_reset	bool	0x0	Will fill in unknown bits in registers at reset with random value using register_reset_data as seed, it overrides scramble_unknowns_at_reset
subcluster0.ras_errselr_undef_if_no_error_records	bool	0x0	If true, ERRSELR_EL1 will be undefined if ERRIDR_EL1 indicates that zero error records are implemented.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-80 Parameters for Subcluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
subcluster0.ras_frac	int	0x0	0, No additional feature implemented. 1, Additional ERXMISC*, ERXPFG* registers and FaultInjection trap from RAS v1.1. implemented
subcluster0.register_reset_data	int	0x0	Data used to fill register bits when they become UNKNOWN at reset.
subcluster0.report_iside_cmo_ifsr	bool	0x1	fault info for an iside cache maintenance operation is reported in the IFSR
subcluster0.reserved_HMC_SSC_PAC_treated_disabled	bool	0x0	When DBG[B|W]CR.{HMC,SSC,PAC} bits configuration is reserved, this parameter controls whether breakpoints/watchpoints are treated as Disabled or not
subcluster0.restriction_on_speculative_execution	int	0x0	Implements the ARMv8.5 security feature (Restrictions on the effects of speculation): 0: No disclosure whether branch targets trained in one context can affect speculative execution in a different context, 1: Branch targets trained in one context cannot affect speculative execution in a different hardware described context (SCXTNUM_ELx not supported), 2: Branch targets trained in one context cannot affect speculative execution in a different hardware described context (SCXTNUM_ELx supported)
subcluster0.rmr_always_implemented	bool	0x0	Always implement RMR_ELx, RMR, or HRMR at the highest implemented exception level, even if that exception level cannot use both AArch32 and AArch64.
subcluster0.rndr_rndrrs_seed	int	0x0	Initial seed for random engine used in RNDR register
subcluster0.scheduler_mode	int	0x0	Control the interleaving of instructions in this processor. 0, default long quantum. 1, low latency mode, short quantum and signal checking. 2, lock-breaking mode, long quantum with additional context switches near load-exclusive instructions.
subcluster0.scramble_unknowns_at_reset	bool	0x1	Will fill in unknown bits in registers at reset with register_reset_data
subcluster0.set_rasv10_for_armv84_and_higher	bool	0x0	ARMv8.4 mandates RAS System Architecture v1.1, but when there are no error records and ARMv8.4-DFE is not implemented then there is no functional difference between the RAS System Architecture v1.0 (that is, the RAS extension as in pre-ARMv8.4 implementations) and the RAS System Architecture v1.1 (also known as ARMv8.4-RAS). This flag if true will set the RAS ID to declare RAS v1.0 rather than RAS v1.1 for ARMv8.4 and higher implementations. If this is set and the core does not conform to the restrictions then this parameter is ignored.
subcluster0.spsr_el3_is_mapped_to_spsr_mon	bool	0x1	Whether SPSR_EL3 is mapped to AArch32 register SPSR_mon
subcluster0.spsr_m4_res0	bool	0x0	Whether SPSR_ELx.M[4] bit should be RES0 for AARCH64 only implementations
subcluster0.stage12_tlb_size	int	0x80	Number of stage1+2 tlb entries.
subcluster0.stage1_tlb_size	int	0x0	Number of stage1 only tlb entries.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-80 Parameters for Subcluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
subcluster0.stage1_walkcache_size	int	0x0	Number of stage1 only walk cache entries.
subcluster0.stage2_tlb_size	int	0x0	Number of stage2 only tlb entries.
subcluster0.stage2_walkcache_size	int	0x0	Number of stage2 only walk cache entries.
subcluster0.statistical_profiling_buffer_alignment	int	0x1	Statistical profiling alignment constraint for sample buffer
subcluster0.statistical_profiling_random_interval_is_separate	bool	0x0	Statistical profiling random interval gets added to the main timer interval(false) or (true) runs as separate timer
subcluster0.statistical_profiling_recommended_min_sampling	int	0x100	Statistical profiling recommended minimum sampling interval
subcluster0.take_ccfail_tsc_trap	bool	0x0	When take_ccfail_undef=1 this prameter controls whether or not an SMC instruction that is trapped by HCR_EL2.TSC but fails its condition code check generates a trap to EL2.
subcluster0.take_ccfail_undef	bool	0x1	UNDEF exception is taken even if condition code check fails
subcluster0.tcr_ps_reserved_value_size	int	0x0	Physical size treated when TCR.(I)PS is programmed with a reserved value. 0, 48 bits. 1, 52 bits. The parameter value is treated 0 if LPA is not supported.
subcluster0.tcr_txsz_undersize_should_fault	bool	0x1	If large VA is not supported, Whether undersized TxSZ value should generate translation fault
subcluster0.tdosa_traps_osdlr_if_no_os_double_lock	bool	0x1	MDCR_EL*.TDOSA enables trap on OSDLR_EL1 and DBGOSDLR when OS double-lock is not implemented.
subcluster0.tidcp_traps_el0_undef_imp_def	bool	0x1	TIDCP has priority over UNDEF for accesses to IMPLEMENTATION DEFINED functionality from EL0
subcluster0.tlb_latency	int	0x0	TLB latency for TA (Timing Annotation), expressed in simulation ticks
subcluster0.trace_physical_registers_when_host_virtualisation_enabled	int	0x0	When host virtualisation is enabled, trace sysreg accesses to physical register accessed (0=disabled, 1=Trace only ELR/SPSR_EL1 as ELR/SPSR_EL2, 2=Trace all redirected registers as physical registers
subcluster0.trap_reserved_group3_id_regs	bool	0x0	Whether setting HCR_EL2.TID3 traps reserved group3 id registers.
subcluster0.treat-dcache-cmos-to-pou-as-nop	bool	0x0	Whether dcache invalidation to the point of unification is required for instruction to data coherence. true - Invalidate operations not required
subcluster0.treat-dcache-invalidate-as-clean-invalidate	bool	0x0	Treat data cache invalidate operations as clean and invalidate.
subcluster0.treat_pld_as_nop	bool	0x0	If true, treat PLD as NOP.
subcluster0.treat_wfi_wfe_as_nop	bool	0x0	If true, never go into wait state for WFI or WFE instructions.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-80 Parameters for Subcluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
subcluster0.truncate_pc_on_illegal_exception_return_to_aarch32	bool	0x1	On Illegal ERET to AArch32, truncate PC to 32-bits
subcluster0.undef_ccsidr2_access_for_unimplemented_aarch32	bool	0x0	Whether access to CCSIDR2 is undef if AArch32 is implemented or not
subcluster0.unification-level	int	0x1	Level of Unification Inner Shareable for the cache hierarchy
subcluster0.unification-uniprocessor-level	int	0x1	Level of Unification Uniprocessor for the cache hierarchy
subcluster0.unpred_nested_virtualization_nv_behaviour	int	0x0	Constrained unpredictable choices for HCR_EL2.NV=0 and HCR_EL2.NV1=1 with respect to nested virtualization - 0, Behave as defined in the sepcification as per bit values - 1, Behave as if HCR_EL2.NV=1 and HCR_EL2.NV1=1 for all purpose other than reading back HCR_EL2.NV - 2, Behave as if HCR_EL2.NV=0 and HCR_EL2.NV1=0 for all purpose other than reading back HCR_EL2.NV1
subcluster0.unpred_vncr_el2_ress_mismatch	int	0x0	Constrained unpredictable choices when bits marked as RESS do not all have the same value for VNCR_EL2 - 0, Generating an EL2 translation regime translation abort on use of the VNCR_EL2 register - 1, Reserved sign exteneded bits of VNCR_EL2 are same as bit[52] or bit[48] based on if large VA is supported or not, for all purposes other than reading back the register
subcluster0.unpred_zero_spsr_btype	bool	0x1	Constrained unpredictable control to make SPSR_ELx.BTYPE 0 instead of PSTATE.BTYPE on synchronous exceptions other than Software Step, PC alignment fault, Instruction Abort, Breakpoint or Address Matching Vector Catch, Illegal Execution State, BRK instruction, Branch Target
subcluster0.unpredictable_exclusive_abort_memtype	int	0x0	Cause MMU abort if exclusive access is not supported in certain memory type (0=exclusives allowed in all memory types, 1=exclusives abort in Device memory types, 2=exclusives abort in any type other than WB inner cacheable)
subcluster0.unpredictable_hvc_behaviour	int	0x0	HVC unpredictable behaviour. 0, UNDEF. 1, NOP.
subcluster0.unpredictable_smc_behaviour	int	0x0	SMC unpredictable behaviour. 0, UNDEF. 1, NOP.
subcluster0.unsupported_atomic_fault_type	int	0x0	Type of fault reported on unsupported atomic access. 0 = external abort if any reported by interconnect, 1 = precise unsupported atomic fault, 2 = precise external abort, 3 = imprecise external abort.
subcluster0.use_Xt_as_LDG_STG_input	bool	0x1	Use new MTE Intructions formats for LDG/ST(Z)(2)G, which use Xt as an input
subcluster0.use_mte_eac_02_instructions_encoding	bool	0x1	Use new MTE Intructions encoding since MTE spec EAC 0.2 in ARMv8.5.
subcluster0.use_mte_eac_08_tfsr_encoding	bool	0x1	Use new MTE TFSR_ELx encodings since MTE spec EAC 0.8 in ARMv8.5.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-80 Parameters for Subcluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
subcluster0.use_rosetta_disass	int	0x1	Use Rosetta disassembly library. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
subcluster0.use_tlb_contig_hint	bool	0x0	Translation table entries with the contiguous hint bit set generate large TLB entries.
subcluster0.walk_cache_latency	int	0x0	Walk cache latency for TA (Timing Annotation), expressed in simulation ticks
subcluster0.warn_unpredictable_in_v7	bool	0x1	If true, behaviour which is unpredictable in V7 yet is predictable in V8 will produce a warning
subcluster0.watchpoint-log2secondary_restriction	int	0x0	log2 size of secondary restriction of FAR/EDWAR possible values on watchpoint hit for load/store operations.
subcluster0.wfe_wakeup_delay	int	0x0	Configure WFE wakeup delay in CPU cycles
subcluster0.wfi_wakeup_delay	int	0x0	Configure WFI wakeup delay in CPU cycles
subcluster0.wnr_is_read_for_s2f_on_s1_atomic_instr_fault	bool	0x0	Whether WnR is 0 for stage2 fault on stage1 for atomic instructions
subcluster0.wnr_is_read_for_s2f_on_s1_dbm_update	bool	0x0	Whether WnR is 0 for stage2 fault on stage1 descriptor dbm update

Table 3-81 Parameters for Cluster_ARMAEMv8-A_MP

Name	Type	Default value	Description
AIDR	int	0x0	Value of AIDR_EL1 register.
AMPIDR	int	0x400bbb000	Value of AMU Peripheral Identification Register
BPIMVA_causes_translation_lookup	bool	0x0	Do a translation when BPIMVA instruction is executed (which may cause a translation fault).
CCSIDR-L1D_override	int	0x0	If nonzero, override the value presented in CCSIDR for L1D (this is cosmetic and does not affect cache behaviour).
CCSIDR-L1I_override	int	0x0	If nonzero, override the value presented in CCSIDR for L1I (this is cosmetic and does not affect cache behaviour).
CCSIDR-L2_override	int	0x0	If nonzero, override the value presented in CCSIDR for L2 (this is cosmetic and does not affect cache behaviour).
CCSIDR-L3_override	int	0x0	If nonzero, allow L3 selection in CSSELR and present this value in CCSIDR (this is cosmetic and does not affect cache behaviour).
CPUCFR	int	0x0	Value of CPU Configuration Register
CTIPIDR	int	0x0	If non-zero, override the CTI Peripheral Identification Register
DBGPIDR	int	0x0	If non-zero, override the Debug Peripheral Identification Register
ERXMISC0_mask	int	0x0	Write Mask for ERXMISC0 RAS Register
GMID-log2-block-size	int	0x4	Log2 of the block size accessed by STGM/LDGM/STZGM instructions

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-81 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
ISV_set_to_0_for_stage2_synch_external_abort	bool	0x0	Whether ESR_EL2.ISV is set to 0 on stage 2 synchronous external aborts
MIDR	int	0x410fd0f0	Value of MIDR_EL1 register.
NUM_CORES	int	0x1	Number of cores per cluster
PA_SIZE	int	0x28	Physical address range supported. For ARMv8.0 and ARMv8.1 this is limited to 48 bits.
PMUPIDR	int	0x0	If non-zero, override the PMU Peripheral Identification Register
abort_execution_from_device_memory	bool	0x0	Execution from device memory generates a prefetch abort.
advsimd_bf16_support_level	int	0x0	Implement BFloat16 operations from ARMv8.6. AArch64 Advanced SIMD and FP BFloat16 instructions are automatically enabled when has_arm_v8-6 is true. - 0, Not implemented. - 1, AArch64 Advanced SIMD and FP BFloat16 instructions only. - 2, AArch32 Advanced SIMD and VFP BFloat16 instructions only. - 3, Both AArch64 Advanced SIMD and FP and AArch32 Advanced SIMD and VFP BFloat16 instructions.
advsimd_i8mm_support_level	int	0x0	Implement Int8 matrix multiply operations from ARMv8.6. AArch64 Advanced SIMD and FP Int8 matrix multiply instructions are automatically enabled when has_arm_v8-6 is true. - 0, Not implemented. - 1, AArch64 Advanced SIMD and FP Int8 matrix multiply instructions only. - 2, AArch32 Advanced SIMD and VFP Int8 matrix multiply instructions only. - 3, Both AArch64 Advanced SIMD and FP and AArch32 Advanced SIMD and VFP Int8 matrix multiply instructions.
amair_reg_rw_mask	int	0x0	RW mask for implementation-defined registers.
amu_aux_voffset_mask	int	0x0	If ARMv8.6 is implemented, each bit of the field, 0 to 15, when 1 indicates that the corresponding virtual offset register, AMEVCNTVOFF1<n>_EL2, is implemented
amu_has_external_interface	int	0x0	Implement external memory-mapped access to system register of activity monitor unit from ARMv8.4. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
amu_has_sysreg_interface	bool	0x1	Implement system register access to activity monitor unit from ARMv8.4.
amu_reset_domain	int	0x0	Reset domain for activity monitor unit. 0, COLD_RESET. 1, WARM_RESET. 2, NONE.
amu_version	int	0x1	Selects the activity monitor version implemented - 1, AMUv1 for Armv8.4 is implemented. - 2, AMUv1 for Armv8.6 is implemented.
apsr_read_restrict	bool	0x0	At EL0, unknown bits of APSR are RAZ.
arm_v8_7_accelerator_support_level	int	0x0	Implements Armv8.7 accelerator support instructions LD/ST64B, ST64BV, ST64BV0, etc.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-81 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
auxilliary_feature_register0	int	0x0	Value of AFR0 ID register.
cache-log2linelen	int	0x6	Log2 of the cache line length in bytes.
cache_maintenance_hits_watchpoints	bool	0x0	DCIMVA operations executed in AArch32 modes hit watchpoints.
changing_block_size_without_bbm_support	int	0x0	Level of support for changing block size without break-before-make.
clear_IT_when_IL_set	bool	0x0	Clear IT bits when performing a *legal* exception return to AArch32 when IL is set
clear_IT_when_IL_set_explicitly	bool	0x0	Apart from clear_IT_when_IL_set, also clear IT bits when loading CPSR from SPSR/memory and IL == 1 in the value being loaded.
clear_reg_top_eret	int	0x1	Behaviour of the upper 32-bits of the Xn registers when changing between AArch32 state and AArch64 state. 0, upper 32-bits preserved for all registers. 1, upper 32-bits set to 0 for all accessible registers. 2, upper 32-bits set to 0 for a random selection of accessible registers. 3, upper-32-bits set to 0 for registers touched in AArch32
cpi_div	int	0x1	Divider for calculating CPI (Cycles Per Instruction)
cpi_mul	int	0x1	Multiplier for calculating CPI (Cycles Per Instruction)
dc_fault_unaligned_s1_device_s2_fwb	bool	0x0	Whether takes an Alignment Fault caused by the memory type on a DC {ZVA,GZVA,GVA} if the stage 1 memory type is any Device memory type
dcache-hit_latency	int	0x0	L1 D-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
dcache-maintenance_latency	int	0x0	L1 D-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
dcache-miss_latency	int	0x0	L1 D-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
dcache-prefetch_enabled	bool	0x0	Enable simulation of data cache prefetching. This is only used when dcache-state_modelled=true
dcache-read_access_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per access (of size dcache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
dcache-read_bus_width_in_bytes	int	0x8	L1 D-Cache read bus width in bytes used to calculate per-access timing annotations

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-81 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
dcache-read_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per byte accessed.dcache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
dcache-size	int	0x8000	L1 D-Cache size in bytes.
dcache-snoop_data_transfer_latency	int	0x0	L1 D-Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
dcache-ways	int	0x2	L1 D-Cache number of ways (sets are implicit from size).
dcache-write_access_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per access (of size dcache-write_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-write_latency is set. This is only used when dcache-state_modelled=true.
dcache-write_bus_width_in_bytes	int	0x8	L1 D-Cache write bus width in bytes used to calculate per-access timing annotations
dcache-write_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per byte accessed. dcache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
dcimva_requires_s2_write_permissions	bool	0x0	Data-cache invalidate by MVA operations require stage 2 write permission (virtualised AArch32 guest).
dczva_reports_lowest_addr_on_tag_check_fail	bool	0x0	Whether DC ZVA reports lowest address in FAR on tag check fail
delayed_dbgreg_between_secure_views	int	0x1	If delayed_dbgreg is enabled, whether the secure and nonsecure external views require explicit synchronization. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
dic-spi_count	int	0x40	Number of shared peripheral interrupts implemented
disable_sve_plugin	bool	0x0	If true, SVE will not be implemented in this processor even if the plugin is loaded
disable_unknown_update_event_on_reset	bool	0x0	Disables SYSREG_UPDATE event notification on reset for the registers whose bitfields are all reserverd or resets to architecturally unknown value
ecv_support_level	int	0x0	Implement Enhanced Counter Virtualization feature from ARMv8.6. 0, Not supported. 1, fully supported without CNTPOFF. 2, fully supported with CNTPOFF.
el0_can_access_imp_def_functionality	bool	0x0	If not made UNDEF by imp_def_functionality_behaviour, EL0 can access IMPLEMENTATION DEFINED registers and system instructions.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-81 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
el0_el1_only_non_secure	bool	0x0	Secure/non-secure state if EL2 and EL3 are not implemented. 0, secure. 1, non-secure.
enable_tlb_contig_check	bool	0x0	Perform extra pagetable walks to check translation table entries that have the contiguous hint bit set.
enhanced_pac2_level	int	0x0	Implements Enhanced PAC2 from ARMv8.6. This feature is mandatory for ARMv8.6 but can be cherrypicked to a ARMv8.3(or greater) implementation. 0: No EnhancedPAC2, 1: EnhancedPAC2 Only, 2: EnhancedPAC2 with FPAC, 3: EnhancedPAC2 with FPACCombined.
error_record_feature_register	string	""	RAS feature register values. An array of JSON objects. The JSON schema for the array is: [{"ED":0x0,"IMPDEF_3_2":0x0,"UI":0x0,"FI":0x0,"UE":0x0,"CFI":0x0,"CEC":0x0,"RP":0x0,"DUI":0x0,"CEO":0x0,"CI":0x0,"TS":0x0,"INJ":0x0,"FRX":0x0,"UC":0x0,"UEU":0x0,"UER":0x0,"UEO":0x0,"DE":0x0,"CE":0x0,"Visibility":"Core"},other_feature_register_values]. Where ED,UI,FI,CE and UE have valid values betwn 0x0 - 0x3. CFI and DUI have valid values 0x0, 0x2 and 0x3. CEC has valid values 0x0,0x2 or 0x4. RP,CEO,INJ,FRX,UC,UEU,UER,UEO,DE has valid values 0x0 or 0x1. CI and TS has valid values of 0x0, 0x1 and 0x2. Visibility has valid values "Core" or "Cluster"
error_record_feature_register_json_file	string	""	File path to the RAS feature register values as JSON. The file uses the same format as the error_record_feature_register parameter value
exercise_stxr_fail	bool	0x0	Reject a pseudo-random majority of exclusive store instructions
ext_abort_device_GRE_prefetch_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_index, Valid indices in range [0, number_of_error_records).
ext_abort_device_GRE_prefetch_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_device_GRE_read_is_critical	bool	0x0	Critical reporting of device-GRE read external aborts.
ext_abort_device_GRE_read_is_sync	int	0x2	Synchronous reporting of device-GRE read external aborts. 0, asynchronous. 1, synchronous. 2, same as ext_abort_device_read_is_sync.
ext_abort_device_GRE_read_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_device_read_ras_index, Valid indices in range [0, number_of_error_records).
ext_abort_device_GRE_read_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_device_read_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_device_GRE_write_is_critical	bool	0x0	Critical reporting of device-GRE write external aborts

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-81 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
ext_abort_device_GRE_write_is_sync	int	0x2	Synchronous reporting of device-GRE write external aborts. 0, asynchronous. 1, synchronous. 2, same as ext_abort_device_write_is_sync.
ext_abort_device_GRE_write_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_device_write_ras_index, Valid indices in range [0, number_of_error_records).
ext_abort_device_GRE_write_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_device_write_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_device_nGRE_prefetch_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_index, Valid indices in range [0, number_of_error_records).
ext_abort_device_nGRE_prefetch_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_device_nGRE_read_is_critical	bool	0x0	Critical reporting of device-nGRE read external aborts.
ext_abort_device_nGRE_read_is_sync	int	0x2	Synchronous reporting of device-nGRE read external aborts. 0, asynchronous. 1, synchronous. 2, same as ext_abort_device_read_is_sync.
ext_abort_device_nGRE_read_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_device_read_ras_index, Valid indices in range [0, number_of_error_records).
ext_abort_device_nGRE_read_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_device_read_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_device_nGRE_write_is_critical	bool	0x0	Critical reporting of device-nGRE write external aborts
ext_abort_device_nGRE_write_is_sync	int	0x2	Synchronous reporting of device-nGRE write external aborts. 0, asynchronous. 1, synchronous. 2, same as ext_abort_device_write_is_sync.
ext_abort_device_nGRE_write_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_device_write_ras_index, Valid indices in range [0, number_of_error_records).
ext_abort_device_nGRE_write_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_device_write_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_device_prefetch_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_index, Valid indices in range [0, number_of_error_records).
ext_abort_device_prefetch_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-81 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
ext_abort_device_read_is_critical	bool	0x0	Critical reporting of device-nGnRE read external aborts
ext_abort_device_read_is_sync	bool	0x1	Synchronous reporting of device-nGnRE read external aborts
ext_abort_device_read_ras_index	int	0x0	External Aborts are reported in RAS record index specified in this param. Values: Valid indices in range [0, number_of_error_records).
ext_abort_device_read_ras_type	int	0x0	External Aborts are reported as RAS error type specified in this param. Values: 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_device_write_is_critical	bool	0x0	Critical reporting of device-nGnRE write external aborts
ext_abort_device_write_is_sync	bool	0x0	Synchronous reporting of device-nGnRE write external aborts
ext_abort_device_write_ras_index	int	0x0	External Aborts are reported in RAS record index specified in this param. Values: Valid indices in range [0, number_of_error_records).
ext_abort_device_write_ras_type	int	0x0	External Aborts are reported as RAS error type specified in this param. Values: 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_fill_data	int	-0x202020303020203	Returned data, if external aborts are asynchronous
ext_abort_normal_cacheable_read_is_critical	bool	0x0	Critical reporting of normal write-back cacheable-read external aborts
ext_abort_normal_cacheable_read_is_sync	bool	0x1	Synchronous reporting of normal write-back cacheable-read external aborts
ext_abort_normal_cacheable_read_ras_index	int	0x0	External Aborts are reported in RAS record index specified in this param. Values: Valid indices in range [0, number_of_error_records).
ext_abort_normal_cacheable_read_ras_type	int	0x0	External Aborts are reported as RAS error type specified in this param. Values: 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_normal_cacheable_write_is_critical	bool	0x0	Critical reporting of normal write-back cacheable write external aborts
ext_abort_normal_cacheable_write_is_sync	bool	0x0	Synchronous reporting of normal write-back cacheable write external aborts
ext_abort_normal_cacheable_write_ras_index	int	0x0	External Aborts are reported in RAS record index specified in this param. Values: Valid indices in range [0, number_of_error_records).
ext_abort_normal_cacheable_write_ras_type	int	0x0	External Aborts are reported as RAS error type specified in this param. Values: 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-81 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
ext_abort_normal_noncacheable_prefetch_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_index, Valid indices in range [0, number_of_error_records).
ext_abort_normal_noncacheable_prefetch_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_normal_noncacheable_read_is_critical	bool	0x0	Critical reporting of normal noncacheable-read external aborts
ext_abort_normal_noncacheable_read_is_sync	bool	0x1	Synchronous reporting of normal noncacheable-read external aborts
ext_abort_normal_noncacheable_read_ras_index	int	0x0	External Aborts are reported in RAS record index specified in this param. Values: Valid indices in range [0, number_of_error_records).
ext_abort_normal_noncacheable_read_ras_type	int	0x0	External Aborts are reported as RAS error type specified in this param. Values: 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_normal_noncacheable_write_is_critical	bool	0x0	Critical reporting of normal noncacheable write external aborts
ext_abort_normal_noncacheable_write_is_sync	bool	0x0	Synchronous reporting of normal noncacheable write external aborts
ext_abort_normal_noncacheable_write_ras_index	int	0x0	External Aborts are reported in RAS record index specified in this param. Values: Valid indices in range [0, number_of_error_records).
ext_abort_normal_noncacheable_write_ras_type	int	0x0	External Aborts are reported as RAS error type specified in this param. Values: 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_normal_wt_cacheable_prefetch_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_index, Valid indices in range [0, number_of_error_records).
ext_abort_normal_wt_cacheable_prefetch_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_normal_wt_cacheable_read_is_critical	bool	0x0	Critical reporting of normal write-through cacheable-read external aborts
ext_abort_normal_wt_cacheable_read_is_sync	int	0x2	Synchronous reporting of normal write-through read external aborts. 0, asynchronous. 1, synchronous. 2, same as ext_abort_normal_cacheable_read_is_sync.
ext_abort_normal_wt_cacheable_read_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_normal_cacheable_read_ras_index, Valid indices in range [0, number_of_error_records).

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-81 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
ext_abort_normal_wt_cacheable_read_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_normal_cacheable_read_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_normal_wt_cacheable_write_is_critical	bool	0x0	Critical reporting of normal write-through write external aborts
ext_abort_normal_wt_cacheable_write_is_sync	int	0x2	Synchronous reporting of normal write-through write external aborts. 0, asynchronous. 1, synchronous. 2, same as ext_abort_normal_cacheable_write_is_sync.
ext_abort_normal_wt_cacheable_write_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_normal_cacheable_write_ras_index, Valid indices in range [0, number_of_error_records).
ext_abort_normal_wt_cacheable_write_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_normal_cacheable_write_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_prefetch_device_GRE_read_is_critical	bool	0x0	Critical reporting of external aborts generated by device-GRE instruction fetches
ext_abort_prefetch_device_GRE_read_is_sync	int	0x2	Behaviour of external aborts generated by device-GRE instruction fetches. 0, asynchronous abort. 1, synchronous abort. 2, same as ext_abort_prefetch_is_sync.
ext_abort_prefetch_device_nGRE_read_is_critical	bool	0x0	Critical reporting of external aborts generated by device-nGRE instruction fetches
ext_abort_prefetch_device_nGRE_read_is_sync	int	0x2	Behaviour of external aborts generated by device-nGRE instruction fetches. 0, asynchronous abort. 1, synchronous abort. 2, same as ext_abort_prefetch_is_sync.
ext_abort_prefetch_device_read_is_critical	bool	0x0	Critical reporting of external aborts generated by device-nGnRE instruction fetches
ext_abort_prefetch_device_read_is_sync	int	0x2	Behaviour of external aborts generated by device-nGnRE instruction fetches. 0, asynchronous abort. 1, synchronous abort. 2, same as ext_abort_prefetch_is_sync.
ext_abort_prefetch_is_critical	bool	0x0	Critical reporting of external aborts generated by normal writeback cacheable instruction fetches
ext_abort_prefetch_is_sync	bool	0x1	Behaviour of external aborts generated by normal writeback cacheable instruction fetches. 0, asynchronous abort. 1, synchronous abort.
ext_abort_prefetch_noncacheable_read_is_critical	bool	0x0	Critical reporting of external aborts generated by normal noncacheable instruction fetches
ext_abort_prefetch_noncacheable_read_is_sync	int	0x2	Behaviour of external aborts generated by normal noncacheable instruction fetches. 0, asynchronous abort. 1, synchronous abort. 2, same as ext_abort_prefetch_is_sync.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-81 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
ext_abort_prefetch_ras_index	int	0x0	External Aborts are reported in RAS record index specified in this param. Values: Valid indices in range [0, number_of_error_records).
ext_abort_prefetch_ras_type	int	0x0	External Aborts are reported as RAS error type specified in this param. Values: 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_prefetch_so_read_is_critical	bool	0x0	Critical reporting of external aborts generated by device-nGnRnE instruction fetches
ext_abort_prefetch_so_read_is_sync	int	0x2	Behaviour of external aborts generated by device=nGnRnE instruction fetches. 0, asynchronous abort. 1, synchronous abort. 2, same as ext_abort_prefetch_is_sync.
ext_abort_prefetch_wt_cacheable_read_is_critical	bool	0x0	Critical reporting of external aborts generated by normal writethrough cacheable instruction fetches
ext_abort_prefetch_wt_cacheable_read_is_sync	int	0x2	Behaviour of external aborts generated by normal writethrough cacheable instruction fetches. 0, asynchronous abort. 1, synchronous abort. 2, same as ext_abort_prefetch_is_sync.
ext_abort_so_prefetch_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_index, Valid indices in range [0, number_of_error_records).
ext_abort_so_prefetch_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_prefetch_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_so_read_is_critical	bool	0x0	Critical reporting of device-nGnRnE read external aborts
ext_abort_so_read_is_sync	bool	0x1	Synchronous reporting of device-nGnRnE read external aborts
ext_abort_so_read_ras_index	int	0x0	External Aborts are reported in RAS record index specified in this param. Values: Valid indices in range [0, number_of_error_records).
ext_abort_so_read_ras_type	int	0x0	External Aborts are reported as RAS error type specified in this param. Values: 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_so_write_is_critical	bool	0x0	Critical reporting of device-nGnRnE write external aborts
ext_abort_so_write_is_sync	bool	0x1	Synchronous reporting of device-nGnRnE write external aborts
ext_abort_so_write_ras_index	int	0x0	External Aborts are reported in RAS record index specified in this param. Values: Valid indices in range [0, number_of_error_records).
ext_abort_so_write_ras_type	int	0x0	External Aborts are reported as RAS error type specified in this param. Values: 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_ttw_cacheable_read_is_critical	bool	0x0	Critical reporting of TTW cacheable read external aborts
ext_abort_ttw_cacheable_read_is_sync	bool	0x1	Synchronous reporting of TTW cacheable read external aborts

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-81 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
ext_abort_ttw_cacheable_read_ras_index	int	0x0	External Aborts are reported in RAS record index specified in this param. Values: Valid indices in range [0, number_of_error_records).
ext_abort_ttw_cacheable_read_ras_type	int	0x0	External Aborts are reported as RAS error type specified in this param. Values: 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_ttw_noncacheable_read_is_critical	bool	0x0	Critical reporting of TTW noncacheable read external aborts
ext_abort_ttw_noncacheable_read_is_sync	bool	0x1	Synchronous reporting of TTW noncacheable read external aborts
ext_abort_ttw_noncacheable_read_ras_index	int	0x0	External Aborts are reported in RAS record index specified in this param. Values: Valid indices in range [0, number_of_error_records).
ext_abort_ttw_noncacheable_read_ras_type	int	0x0	External Aborts are reported as RAS error type specified in this param. Values: 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
ext_abort_ttw_wt_cacheable_read_is_critical	bool	0x0	Critical reporting of TTW write-through cacheable read external aborts
ext_abort_ttw_wt_cacheable_read_is_sync	int	0x2	Synchronous reporting of TTW write-through cacheable read external aborts. 0, asynchronous. 1, synchronous. 2, same as ext_abort_ttw_cacheable_read_is_sync.
ext_abort_ttw_wt_cacheable_read_ras_index	int	-0x1	External Aborts are reported in RAS record index specified in this param. Values: -1 = Same as ext_abort_ttw_cacheable_read_ras_index, Valid indices in range [0, number_of_error_records).
ext_abort_ttw_wt_cacheable_read_ras_type	int	-0x1	External Aborts are reported as RAS error type specified in this param. Values: -1 = Same as ext_abort_ttw_cacheable_read_ras_type, 0 = NONE, 1 = UC, 2 = UEU, 3 = UEO , 4 = UER, 5 = CE.
fault_on_nT_bit_set	bool	0x1	Whether block translation table entries with the nT bit set should always fault. Only applies when changing_block_size_without_bbm_support_level is 1 or higher.
fault_unalign_to_unsupported_mem	int	0x0	If has_unaligned_single_copy_atomicity is true, whether unaligned A64 atomic, exclusive and acquire/release instructions to non iWB-oWB generate fault. This parameter is a bitmap. Bits 0,1,2 should be set accordingly to enable the fault behaviour. bit 0(LSB): atomic access should fault, bit 1: exclusive access should fault, bit 2: acquire/release should fault.
fault_unaligned_s1_device_s2_fwb	bool	0x0	Whether unaligned access with stage1 Device memory and final memory attribute forced to normal by FWB can generate alignment fault
force_deterministic_irg_tag_generation	bool	0x0	Force the random tag generated by the IRG instruction when GCR_EL1.RRND=1 to equal RGSR_EL1.SEED[3:0] rather than a non-deterministic value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-81 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
fsr_ext_bit_update_kind	int	0x3	Set/Clear DFSR/IFSR EA bit on Synchronous/Async External Aborts. 0: Never Set, 1: Set on Synchronous Ext Aborts 2: Set on Asynchronous Ext Aborts 3: Set on both Sync and Async Ext Aborts.
global_debug_rom.ROMDEVID	int	0x0	Value of Debug Rom Device Identification Register
global_debug_rom.ROMPIDR	int	0x4000 bb000	Value of Debug Rom Peripheral Identification Register
hardware_translation_table_update_implemented	int	0x1	Implement hardware translation table updates from ARMv8.1. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.1 is enabled. - 2, feature is implemented.
has_16bit_asids	bool	0x1	Enable 16-bit ASIDs.
has_16bit_vmids	int	0x1	Implement support for 16-bit VMIDs from ARMv8.1. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.1 is enabled. - 2, feature is implemented.
has_16k_granule	bool	0x0	Implement the 16k LPAE translation granule.
has_4k_granule	bool	0x1	Implement the 4k LPAE translation granule.
has_52bit_address_with_16k	int	0x0	Implements Armv8.7 52-bit IPA/PA support for 16k Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if Armv8.7 is enabled. - 2, feature is implemented.
has_52bit_address_with_4k	int	0x0	Implements Armv8.7 52-bit IPA/PA support for 4k Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if Armv8.7 is enabled. - 2, feature is implemented.
has_64k_granule	bool	0x1	Implement the 64k LPAE translation granule.
has_aarch32_hpd	bool	0x0	If true then hierarchical permission disable is supported in AArch32
has_aarch64	bool	0x1	All implemented exception levels can run in AArch64
has_actlr2	bool	0x0	If true ACLTR2 exists and ACTLR2(NS) is aliased to ACTLR_EL1[63:32]
has_amu	int	0x0	Implement activity monitor functionality from ARMv8.4. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_arm_v8-1	bool	0x0	Implement the ARMv8.1 Extension.
has_arm_v8-2	bool	0x0	Implement the ARMv8.2 Extension.
has_arm_v8-3	bool	0x0	Implement the ARMv8.3 Extension.
has_arm_v8-4	bool	0x0	Implement the ARMv8.4 Extension.
has_arm_v8-5	bool	0x0	Implement the ARMv8.5 Extension.
has_arm_v8-6	bool	0x0	Implement the ARMv8.6 Extension.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-81 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
has_arm_v8-7	bool	0x0	Implement the Armv8.7 Extension. This feature is incomplete and under development.
has_at_with_pan	int	0x1	Implement new AT instructions with PAN support. Possible values of this parameter are: - 1, feature is implemented if ARMv8.2 is enabled. - 2, feature is implemented.
has_axflag_xaflag	int	0x1	Implement flag manipulation instructions (AXFlag, XAFlag) from ARMv8.5. Possible values of this parameter are: - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
has_axflag_xaflag_frint	int	0x1	Implement flag manipulation instructions (AXFlag, XAFlag) and floating-point rounding to int instructions (FRINT[32|64][X|Z]) from ARMv8.5. If this parameter is enabled, it also enables both has_axflag_xaflag and has_frint. If support for only one of the features is needed, please use the individual parameters and do not enable this one. Possible values of this parameter are: - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
has_branch_target_exception	int	0x0	Implement Branch target identification mechanism from ARMv8.5. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
has_ccidx	bool	0x0	Implement the ARMv8.3 CCSIDR Extension. Extending the ccsidr number of sets.
has_cfinv_rmif_setf	int	0x1	Implement flag manipulation (CFINV, RMIF, SETF8, SETF16) instructions from ARMv8.4. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_coherent_icache	bool	0x0	Whether icache invalidation to the point of unification is required for instruction to data coherence. true - Invalidate operations not required
has_common_not_private_translations	int	0x1	Implement the TTBRn_ELx.CnP (Common not Private) controls from ARMv8.2. Possible values of this parameter are: - 1, feature is implemented if ARMv8.2 is enabled. - 2, feature is implemented.
has_complex_number	int	0x1	Implement ARMv8.3 complex number support, Multiply Accumulate and Add instructions. Possible values of this parameter are: - 1, feature is implemented if ARMv8.3 is enabled. - 2, feature is implemented.
has_cvadp_support	int	0x0	Implement instruction to support cache clean by deep persistence (DC CVADP) from ARMv8.5, can be selected for core implemented on any arch version starting ARMv8.2 Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.2 is enabled. - 2, feature is implemented.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-81 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
has_data_alignment_flag	int	0x0	Implement non-optimal misalignment flag for PMU/SPE from ARMv8.5 Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
has_delayed_ctireg	bool	0x0	Delay the functional effect of CTI register writes until ISB or implicit barrier.
has_delayed_dbgreg	bool	0x0	Delay the functional effect of external debug register writes until ISB or implicit barrier.
has_delayed_sysreg	bool	0x0	Delay the functional effect of system register writes until ISB or implicit barrier.
has_delayed_wfe_trap	int	0x0	Implements Configurable Delayed WFE trapping from ARMv8.6 Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.6 is enabled. - 2, feature is implemented.
has_dgh	int	0x1	Implements Data Gathering Hint instruction from ARMv8.6 Possible values of this parameter are: - 1, feature is implemented if ARMv8.6 is enabled. - 2, feature is implemented.
has_dot_product	int	0x1	Implement the dot product (UDOT, SDOT) instructions from ARMv8.4. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_e0pd	int	0x1	Implement ARMv8.5 feature to prevent unprivileged access to one half of the memory Possible values of this parameter are: - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
has_el2	bool	0x1	Implements EL2
has_el3	bool	0x1	Implements EL3
has_enhanced_pac	bool	0x0	If pointer authentication is enabled then implement enhanced PAC.
has_enhanced_pan	int	0x1	Implements Armv8.7 Enhanced PAN feature Possible values of this parameter are: - 1, feature is implemented if Armv8.7 is enabled. - 2, feature is implemented.
has_ets	bool	0x0	Whether Enhanced Translation Synchronization is supported
has_extended_recp_estimate	int	0x0	Implements increased precision of reciprocal instructions.) Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if Armv8.7 is enabled. - 2, feature is implemented.
has_fgt	int	0x1	Implements Fine-grained Virtualization Traps extension from ARMv8.6 Possible values of this parameter are: - 1, feature is implemented if ARMv8.6 is enabled. - 2, feature is implemented.
has_fp16	int	0x1	Implement the half-precision floating-point data processing instructions from ARMv8.2. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.2 is enabled. - 2, feature is implemented.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-81 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
has_fp16_fmlal	int	0x1	Implement the New Floating Point Multiplication Variant (FP16 FMLAL, FMLSL) instructions from ARMv8.4. Only supported if has_fp16=0x1. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_frint	int	0x1	Implement floating-point rounding to int instructions (FRINT[32|64][X|Z]) from ARMv8.5. Possible values of this parameter are: - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
has_generic_authentication	int	0x1	Implement ARMv8.3 generic authentication. Possible values of this parameter are: - 1, feature is implemented if ARMv8.3 is enabled. - 2, feature is implemented.
has_guest_translation_granule	int	0x1	Implement mechanism for guest translation granule identification from ARMv8.5, ID values determined by stage1 granule configuration parameters Possible values of this parameter are: - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
has_hardware_translation_table_update	int	0x2	Type of hardware translation table supported (when enabled by hardware_translation_table_update_implemented). 0, not implemented. 1, access bit updates implemented. 2, access bit updates and dirty bit mechanism implemented.
has_hcrx_el2	int	0x1	Implements new HCRX_EL2 id register from Armv8.7 Possible values of this parameter are: - 1, feature is implemented if Armv8.7 is enabled. - 2, feature is implemented.
has_id_reg_read	int	0x1	Implement read access to the ID registers (ESR_ELx.EC=0x18) Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_iesb	int	0x1	Implement support for implicit error sync event from ARMv8.2. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.2 is enabled. - 2, feature is implemented.
has_jscvt	int	0x1	Implement ARMv8.3 javascript Floating-point to Integer conversion instruction. Possible values of this parameter are: - 1, feature is implemented if ARMv8.3 is enabled. - 2, feature is implemented.
has_large_system_ext	bool	0x0	Implement the ARMv8 Large System Extensions.
has_large_va	int	0x0	Implement support for the extended 52-bit virtual addresses from ARMv8.2. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.2 is enabled. - 2, feature is implemented.
has_ldapur_stlur	int	0x1	Implement support for LDAPR and STLR instructions with immediate offsets from ARMv8.4. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-81 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
has_ldm_stm_ordering_control	int	0x0	Implement the SCTLR_ELx.LSMAOE (Load/Store Multiple Atomicity and Ordering Enable) and SCTLR_ELx,nTLSMD (no Trap Load/Store Multiple to Device) controls from ARMv8.2. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.2 is enabled. - 2, feature is implemented.
has_lrcpc	bool	0x0	If true then it support the RCpc feature (ARMv8.3)
has_mpam	int	0x0	Implement ARMv8.4 MPAM Registers and associated functionality. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_mt_pmu_disable_feature	int	0x0	Implements Multi-threading PMU disable extension from ARMv8.6 Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.6 is enabled. - 2, feature is implemented.
has_mte_ctrl_bits_stateful	bool	0x0	if memory_tagging_support_level == 1, Whether mte specific control bits in system registers are stateful
has_nested_virtualization	int	0x1	Implement ARMv8.3 nested virtualization. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.3 is enabled. - 2, feature is implemented.
has_no_os_double_lock	int	0x0	Do not implement the OS double-lock. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_non_context_synchronizing_exception_controls	int	0x1	Implement cosmetic controls for whether exception entry and exit are context synchronizing events (SCTLR_ELx.{EIS,EOS}) from ARMv8.5. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
has_par_bit10_razwi	bool	0x0	Whether PAR_EL1[10] is RAZ/WI
has_pointer_authentication	int	0x1	Implement ARMv8.3 pointer authentication. Possible values of this parameter are: - 1, feature is implemented if ARMv8.3 is enabled. - 2, feature is implemented.
has_prediction_invalidation_instructions	int	0x1	Implement execution and data prediction invalidation from ARMv8.5 Possible values of this parameter are: - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
has_pstate_dit	int	0x1	Implement timing insensitivity of data processing instructions from ARMv8.4. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_pstate_pan	int	0x1	Implement the PSTATE.PAN (Privileged Access Never) control from ARMv8.1 Possible values of this parameter are: - 1, feature is implemented if ARMv8.1 is enabled. - 2, feature is implemented.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-81 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
has_pstate_uao	int	0x1	Implement the PSTATE.UAO (User Access Override) control from ARMv8.2. Possible values of this parameter are: - 1, feature is implemented if ARMv8.2 is enabled. - 2, feature is implemented.
has_ras_armv84_extension	int	0x1	Implement ARMv8.4 RAS Extension. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_ras_critical_error	int	0x0	[DEPRECATED: Set CI field on first register in error_record_feature_register JSON instead] ARMv8.4 AArch64 RAS Critical Error is implemented or not. 0 - Feature Not Supported, 1 - Feature always enabled, 2 - Feature is controllable.
has_ras_double_fault	int	0x1	Implement ARMv8.4 RAS Doublt Fault Extension. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_ras_fault_injection	int	0x0	[DEPRECATED: Set INJ field on first register in error_record_feature_register JSON instead] Implement ARMv8.4 Standard Fault Injection mechanism. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_ras_timestamp	int	0x0	[DEPRECATED: Set TS field on first register in error_record_feature_register JSON instead] ARMv8.4 AArch64 RAS Timestamp register is implemented or not. 0 - No Timestamp is recorded, 1 - Generic Timer timestamp is recorded, 2 - IMP DEF timestamp is recorded.
has_restriction_on_speculative_data_loaded	int	0x1	Implements the ARMv8.5 security feature (Restrictions on the effects of speculation) Possible values of this parameter are: - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
has_rndr	int	0x0	Implement random number instructions to read from RNDR and RNDRSS random number registers from ARMv8.5. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
has_rounding_doubling_multiply_add_subtract	int	0x1	Implement the rounding doubling multiply add and subtract instructions from ARMv8.1 Possible values of this parameter are: - 1, feature is implemented if ARMv8.1 is enabled. - 2, feature is implemented.
has_secure_el2	int	0x1	Implement support for Secure EL2 Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_self_hosted_trace_extension	int	0x1	Implement support for the Self-hosted Trace Extensions from ARMv8.4. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-81 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
has_small_page_table	int	0x1	Implement small page table support which increases the maximum value of TxSZ field from ARMv8.4. Note: will be unimplemented only if both has_small_page_table=0x0 and has_secure_el2=0x0. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_speculation_barrier_inst	int	0x1	Implement speculation barrier instruction (SB) from ARMv8.5 Possible values of this parameter are: - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
has_stage2_fwb	int	0x1	Implement HCR_EL2.FWB, stage 2 control of memory types and cacheability Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_stage2_xnx	int	0x1	Implement the extended XN[1:0] stage 2 control from ARMv8.2. Possible values of this parameter are: - 1, feature is implemented if ARMv8.2 is enabled. - 2, feature is implemented.
has_stage2_xnx_in_aarch32	bool	0x1	Implement the extended XN[1:0] stage 2 control from ARMv8.2 for Aarch32.
has_statistical_profiling	bool	0x1	Whether Statistical Based Profiling is implemented
has_supersections	bool	0x1	Whether VMSAv8-32 supersection to support more than 32-bit PA using short descriptor is implemented.
has_tlb_conflict_abort	bool	0x0	Detected inconsistent TLB content generate aborts.
has_tlbi_range	int	0x1	Implement support for TLB Range Maintenance instructions (TLBI RVAE1, etc) from ARMv8.4. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_tlbi_to_outer_shareable	int	0x1	Implement support for TLB Maintenance instructions that extend to the Outer Shareable domain (TLBI VAE1OS, etc) from ARMv8.4. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_tlbi_ttl	int	0x1	Implement support for the TTL level hint in by-address TLB Maintenance instructions from ARMv8.4. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_unaligned_single_copy_atomicity	int	0x1	Implement support for SCTLR_ELx.nAA from ARMv8.4, and A64 atomic, exclusive and acquire/release instructions accessing unaligned bytes inside a 16byte window will not generate alignment fault. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_v8_4_debug_extension	int	0x1	Implement ARMv8.4 debug extensions Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-81 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
has_v8_4_pmu_extension	int	0x1	Implement PMU extension from ARMv8.4. Possible values of this parameter are: - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_v8_5_debug_over_power_down	int	0x0	Implement ARMv8.5 Debug over powerdown Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
has_v8_5_pmu_extension	int	0x1	Implement PMU extension from ARMv8.5. Possible values of this parameter are: - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
has_v8_5_spe_extension	int	0x1	Implement SPE extension from ARMv8.5 Possible values of this parameter are: - 1, feature is implemented if ARMv8.5 is enabled. - 2, feature is implemented.
has_v8_6_pmu_events	int	0x1	Implements PMU events from ARMv8.6 Possible values of this parameter are: - 1, feature is implemented if ARMv8.6 is enabled. - 2, feature is implemented.
has_v8_7_fp_enhancements	int	0x1	Implements the Floating Point enhancements from Armv8.7 (introduces FPCR.FIZ/AH/NEP, etc.) Possible values of this parameter are: - 1, feature is implemented if Armv8.7 is enabled. - 2, feature is implemented.
has_v8_7_pmu_events	int	0x1	Implement PMU events from ARMv8.7. Possible values of this parameter are: - 1, feature is implemented if Armv8.7 is enabled. - 2, feature is implemented.
has_v8_7_pmu_extension	int	0x1	Implement PMU extension from ARMv8.7. Possible values of this parameter are: - 1, feature is implemented if Armv8.7 is enabled. - 2, feature is implemented.
has_v8_7_spe_extension	int	0x1	Implement SPE extension from ARMv8.7. Possible values of this parameter are: - 1, feature is implemented if Armv8.7 is enabled. - 2, feature is implemented.
has_vncr_el2	int	0x1	Implement support for nested virtualization enhancements from ARMv8.4. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
has_wfet_and_wfit	int	0x1	Implements WFE and WFI with Timeout from Armv8.7 Possible values of this parameter are: - 1, feature is implemented if Armv8.7 is enabled. - 2, feature is implemented.
has_writebuffer	bool	0x0	Implement write accesses buffering before L1 cache. May affect ext_abort behaviour.
has_xs	int	0x1	Implements Armv8.7 XS, TLBInXS, DSBnXS instruction Possible values of this parameter are: - 1, feature is implemented if Armv8.7 is enabled. - 2, feature is implemented.
icache-hit_latency	int	0x0	L1 I-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when icache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-81 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
icache-log2linelen	int	0x0	If nonzero, Log2 of the instruction cache line length in bytes (valid values in range 4-8). Otherwise the value of cache-log2linelen is used.
icache-maintenance_latency	int	0x0	L1 I-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when icache-state_modelled=true.
icache-miss_latency	int	0x0	L1 I-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when icache-state_modelled=true.
icache-nprefetch	int	0x1	Number of next sequential instruction cache lines to prefetch. This is only used when icache-prefetch_enabled=true.
icache-prefetch_enabled	bool	0x0	Enable simulation of instruction cache prefetching. This is only used when icache-state_modelled=true.
icache-prefetch_level	int	0x0	0 based cache level at which instructions are pre-fetched. This is only used when icache-prefetch_enabled=true.
icache-read_access_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per access (of size icache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if icache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when icache-state_modelled=true.
icache-read_bus_width_in_bytes	int	0x8	L1 I-Cache read bus width in bytes used to calculate per-access timing annotations
icache-read_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per byte accessed.icache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when icache-state_modelled=true.
icache-size	int	0x8000	L1 I-Cache size in bytes.
icache-ways	int	0x2	L1 I-Cache number of ways (sets are implicit from size).
ignore_DBGPRCR_CWRR	bool	0x0	Ignore writes to the deprecated DBGPRCR.CWRR bit.
ignore_access_flag_update_by_at_ops	bool	0x0	If true, AT operations do not update access flag
ignore_large_address_top_bits_in_page_walk	bool	0x0	Whether page table bits [15:12] are ignored if PA_SIZE < 52 and output address is configured < 52 with large page
imp_def_functionality_behaviour	int	0x0	Behaviour of IMPLEMENTATION DEFINED registers and system instructions. 0, UNDEF. 1, RAZ/WI.
impdef_regs_and_unpred_from_implementation	string	""	Configure implementation defined registers and unpredictable behaviour to match the specified implementation. Requires a license for the selected implementation model. Use ARM_Cortex-A<num> or ARM_<codename> for licensed pre-release cores.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-81 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
independent_cache_control_traps	int	0x0	Implement Independent Cache Control traps from ARMv8.5. 0, NO_SUPPORT. 1, SUPPORTED_BUT_NOT_FOR_TLB_MAINTENANCE_INSTRUCTIONS. 2, FULL_SUPPORT.
insert_iesb_before_exception	bool	0x0	If true then inserts an IESB before taking with Exception otherwise has no effect and IESB is taken after PState is changed due to the Exception.
instruction_tlb_size	int	0x0	Number of stage1+2 itlb entries (or 0 for unified ITLB+DTLB)
is_mt_res0	bool	0x0	If ARMv8.6 is not implemented, and PMUv3 is implemented, this parameter controls whether PMEVTYPER<n>.MT bit is RES0 or RW. For other implementations, this parameter has no effect
is_uniprocessor	bool	0x0	Value for the U bit in MPIDR. true disables L1 cache coherency protocols
l2cache-hit_latency	int	0x0	L2 Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
l2cache-maintenance_latency	int	0x0	L2 Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
l2cache-miss_latency	int	0x0	L2 Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
l2cache-read_access_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
l2cache-read_bus_width_in_bytes	int	0x8	L2 Cache read bus width in bytes used to calculate per-access timing annotations
l2cache-read_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per byte accessed.l2cache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
l2cache-size	int	0x80000	L2 Cache size in bytes.
l2cache-snoop_data_transfer_latency	int	0x0	L2 Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
l2cache-snoop_issue_latency	int	0x0	L2 Cache timing annotation latency for snoop accesses issued by this cache in total ticks. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-81 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
l2cache-ways	int	0x10	L2 Cache number of ways (sets are implicit from size).
l2cache-write_access_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-write_latency is set. This is only used when dcache-state_modelled=true.
l2cache-write_bus_width_in_bytes	int	0x8	L2 Cache write bus width in bytes used to calculate per-access timing annotations
l2cache-write_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per byte accessed. l2cache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
l3cache-mpamf.bwa_width_ns	int	0x10	L3 Cache width of MPAM bandwidth allocation fields for non-secure accesses.
l3cache-mpamf.bwa_width_s	int	0x10	L3 Cache width of MPAM bandwidth allocation fields for secure accesses.
l3cache-mpamf.cmax_width_ns	int	0x0	L3 Cache number of fractional bits in MPAM cache capacity partition control for non-secure accesses. Only the register interface is implemented - the control is NOT FUNCTIONAL.
l3cache-mpamf.cmax_width_s	int	0x0	L3 Cache number of fractional bits in MPAM cache capacity partition control for secure accesses. Only the register interface is implemented - the control is NOT FUNCTIONAL.
l3cache-mpamf.cpbm_width_ns	int	0x0	L3 Cache width of MPAM cache portion bitmap for non-secure accesses. If 0, the feature is not implemented, and all ways are available.
l3cache-mpamf.cpbm_width_s	int	0x0	L3 Cache width of MPAM cache portion bitmap for secure accesses. If 0, the feature is not implemented, and all ways are available.
l3cache-mpamf.csu_num_mon_ns	int	0x0	L3 Cache number of MPAM cache storage usage monitors for non-secure accesses.
l3cache-mpamf.csu_num_mon_s	int	0x0	L3 Cache number of MPAM cache storage usage monitors for secure accesses.
l3cache-mpamf.has_prop_ns	bool	0x0	Enable memory bandwidth proportional stride control for non-secure accesses. Only the register interface is implemented - the control is NOT FUNCTIONAL.
l3cache-mpamf.has_prop_s	bool	0x0	Enable memory bandwidth proportional stride control for secure accesses. Only the register interface is implemented - the control is NOT FUNCTIONAL.
max_32bit_el	int	0x3	Maximum exception level supporting AArch32 modes.
memory.acp.AxCACHE_mask	int	0x0	Used with memory.acp.AxCACHE_pattern to define which memory types the ACP port accepts. All transactions which do not satisfy (AxCACHE & mask) == pattern will abort.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-81 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
memory.acp.AxCACHE_pattern	int	0x0	Used with memory.acp.AxCACHE_mask to define which memory types the ACP port accepts. All transactions which do not satisfy (AxCACHE & mask) == pattern will abort.
memory.l2_cache.is_inner_cacheable	bool	0x1	L2 cache obeys inner cacheable attributes (rather than outer cacheable attributes)
memory.l2_cache.is_inner_shareable	bool	0x1	L2 cache obeys inner shareable attributes (rather than outer sharaable attributes)
memory_tagging_support_level	int	0x0	Specify the memory tagging extension support level: 0, not implemented. 1, instructions and registers only are implemented. 2, implemented. 3, implemented with asymmetric handling of exceptions
mixed_endian	int	0x1	Implement support for mixed endianness. 0, not supported. 1, supported at all exception levels. 2, supported at EL0 only.
mpam_force_ns_rao	bool	0x0	Whether MPAM3_EL3.FORCE_NS bit is RAO/WI.
mpam_frac	int	0x0	MPAM fractional revision number in ID_AA64PFR1_EL1.MPAM_frac field
mpam_has_hcr	bool	0x0	MPAM Whether MPAMIDR_EL1 HAS_HCR bit is set or clear
mpam_max_partid	int	0xffff	MPAM Maximum PARTID Supported
mpam_max_pmg	int	0xff	MPAM Maximum PMG Supported
mpam_max_vpmr	int	0x0	MPAM Maximum VPMR Supported
mpamidr_has_force_ns	int	0x0	Whether MPAMIDR_EL1.HAS_FORCE_NS bit is set or clear. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.6 is enabled. - 2, feature is implemented.
mpamidr_has_sdeflt	int	0x0	Whether MPAMIDR_EL1.HAS_SDEFLT bit is set or clear. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.6 is enabled. - 2, feature is implemented.
mpamidr_has_tidr	int	0x0	Whether MPAMIDR_EL1.HAS_TIDR bit is set or clear Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.6 is enabled. - 2, feature is implemented.
mvbar_reset_is_rvbar	bool	0x1	If true then the reset value of MVBAR is RVBAR, if false the reset value is UNKNOWN.
non_secure_vgic_alias_when_ns_only	int	0x0	If ! has_el3 and only non-secure side exists, then the normal position of the VGIC is a secure alias. If this parameter is non-zero then in addition a non-secure alias of the VGIC will be placed at this position (aligned to 32 KB).
num_loregion_descriptors	int	0x0	Number of Limited Ordering Region descriptors implemented (if ARM v8.1 extensions are implemented)
num_loregions	int	0x0	Number of Limited Ordering Regions implemented excluding background region (if ARM v8.1 extensions are implemented)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-81 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
number_of_error_records	int	0x0	Cores Number of Error records supported for RAS
optimal_alignment_size	int	0x1	Alignment boundary which does not incur additional performance penalty from ARMv8.5. - 1, architectural misalignment is used to set PMU event LDST_ALIGN_LAT and SPE event E[11] - 2, access crossing 4 byte boundary is used to set PMU event LDST_ALIGN_LAT and SPE event E[11] - 3, access crossing 8 byte boundary is used to set PMU event LDST_ALIGN_LAT and SPE event E[11] ... - 12, access crossing 4 Kbyte boundary is used to set PMU event LDST_ALIGN_LAT and SPE event E[11]
output_attributes	string	"ExtendedID[62:55]=MPAM_PMG, ExtendedID[54:39]=MPAM_PARTID, ExtendedID[38]=MPAM_NS"	User-defined transform to be applied to bus attributes like MasterID, ExtendedID or UserFlags. Currently, only works for MPAM Attributes encoding into bus attributes.
page_based_hardware_attributes	int	0x0	Implement the page based hardware attributes from ARMv8.2. This parameter indicates which page table bits are available for hardware, where bits[3:0] correspond to PTE[62:59] and to TCR_ELx.HWUnyy.
pmmir_el1_bus_slots	int	0x0	Largest value by which BUS_ACCESS can increment over BUS_CYCLES cycles. From v8.7 PMU extension
pmmir_el1_bus_width	int	0x0	Width, in bytes, of accesses counted by BUS_ACCESS. From v8.7 PMU extension
pseudo_fault_generation_feature_register	string	""	ARMv8.4 Standard Pseudo-fault generation feature register values. JSON schema for the parameter value is: [{"OF":false,"UC":false,"UEU":false,"UER":false,"UEO":false,"DE":false,"CE":0x0,"CI":false,"ER":false,"PN":false,"AV":false,"MV":false,"SYN":false,"R":false},other_psuedo-fault_generating_features_register_values]. Where OF, UC, UEU, UER, UEO, DE, CI, ER, PN, AV, MV, SYN, and R have valid false(NOT_SUPPORTED) and true(FEATURE_CONTROLLABLE), where CE can have 0(NOT_SUPPORTED), 1(NONSPECIFIC_CE_SUPPORTED) and 3(TRANSIENT_OR_PERSISTENT_CE_SUPPORTED). Effective only when ERXFR's INJ field allows it or has_ras_fault_injection is true.
pstate_ssbs_type	int	0x0	Implement speculative store bypass safe feature from ARMv8.5. 0, Not supported. 1, Supported without MSR/MRS access to SSBS. 2, fully supported.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-81 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
ptw_latency	int	0x0	Page table walker latency for TA (Timing Annotation), expressed in simulation ticks
randomize_unknowns_at_reset	bool	0x0	Will fill in unknown bits in registers at reset with random value using register_reset_data as seed, it overrides scramble_unknowns_at_reset
ras_errselr_undef_if_no_error_records	bool	0x0	If true, ERRSELR_EL1 will be undefined if ERRIDR_EL1 indicates that zero error records are implemented.
ras_frac	int	0x0	0, No additional feature implemented. 1, Additional ERXMISC*, ERXPFG* registers and FaultInjection trap from RAS v1.1. implemented
register_reset_data	int	0x0	Data used to fill register bits when they become UNKNOWN at reset.
restriction_on_speculative_execution	int	0x0	Implements the ARMv8.5 security feature (Restrictions on the effects of speculation): 0: No disclosure whether branch targets trained in one context can affect speculative execution in a different context, 1: Branch targets trained in one context cannot affect speculative execution in a different hardware described context (SCXTNUM_ELx not supported), 2: Branch targets trained in one context cannot affect speculative execution in a different hardware described context (SCXTNUM_ELx supported)
rmr_always_implemented	bool	0x0	Always implement RMR_ELx, RMR, or HRMR at the highest implemented exception level, even if that exception level cannot use both AArch32 and AArch64.
rndr_rndrrs_seed	int	0x0	Initial seed for random engine used in RNDR register
scheduler_mode	int	0x0	Control the interleaving of instructions in this processor. 0, default long quantum. 1, low latency mode, short quantum and signal checking. 2, lock-breaking mode, long quantum with additional context switches near load-exclusive instructions.
scramble_unknowns_at_reset	bool	0x1	Will fill in unknown bits in registers at reset with register_reset_data
set_rasv10_for_armv84_and_higher	bool	0x0	ARMv8.4 mandates RAS System Architecture v1.1, but when there are no error records and ARMv8.4-DFE is not implemented then there is no functional difference between the RAS System Architecture v1.0 (that is, the RAS extension as in pre-ARMv8.4 implementations) and the RAS System Architecture v1.1 (also known as ARMv8.4-RAS). This flag if true will set the RAS ID to declare RAS v1.0 rather than RAS v1.1 for ARMv8.4 and higher implementations. If this is set and the core does not conform to the restrictions then this parameter is ignored.
spsr_el3_is_mapped_to_spsr_mon	bool	0x1	Whether SPSR_EL3 is mapped to AArch32 register SPSR_mon
spsr_m4_res0	bool	0x0	Whether SPSR_ELx.M[4] bit should be RES0 for AARCH64 only implementations
stage12_tlb_size	int	0x80	Number of stage1+2 tlb entries.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-81 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
stage1_tlb_size	int	0x0	Number of stage1 only tlb entries.
stage1_walkcache_size	int	0x0	Number of stage1 only walk cache entries.
stage2_tlb_size	int	0x0	Number of stage2 only tlb entries.
stage2_walkcache_size	int	0x0	Number of stage2 only walk cache entries.
statistical_profiling_buffer_alignment	int	0x1	Statistical profiling alignment constraint for sample buffer
statistical_profiling_random_interval_is_separate	bool	0x0	Statistical profiling random interval gets added to the main timer interval(false) or (true) runs as separate timer
statistical_profiling_recommended_min_sampling	int	0x100	Statistical profiling recommended minimum sampling interval
take_ccfail_tsc_trap	bool	0x0	When take_ccfail_undef=1 this prameter controls whether or not an SMC instruction that is trapped by HCR_EL2.TSC but fails its condition code check generates a trap to EL2.
take_ccfail_undef	bool	0x1	UNDEF exception is taken even if condition code check fails
tcr_ps_reserved_value_size	int	0x0	Physical size treated when TCR.(I)PS is programmed with a reserved value. 0, 48 bits. 1, 52 bits. The parameter value is treated 0 if LPA is not supported.
tcr_txsz_undersize_should_fault	bool	0x1	If large VA is not supported, Whether undersized TxSZ value should generate translation fault
tdosa_traps_osdlr_if_no_os_double_lock	bool	0x1	MDCR_EL*.TDOSA enables trap on OSDLR_EL1 and DBGOSDLR when OS double-lock is not implemented.
tidcp_traps_el0_undef_imp_def	bool	0x1	TIDCP has priority over UNDEF for accesses to IMPLEMENTATION DEFINED functionality from EL0
tlb_latency	int	0x0	TLB latency for TA (Timing Annotation), expressed in simulation ticks
trace_physical_registers_when_host_virtualisation_enabled	int	0x0	When host virtualisation is enabled, trace sysreg accesses to physical register accessed (0=disabled, 1=Trace only ELR/SPSR_EL1 as ELR/SPSR_EL2, 2=Trace all redirected registers as physical registers
treat-dcache-cmos-to-pou-as-nop	bool	0x0	Whether dcache invalidation to the point of unification is required for instruction to data coherence. true - Invalidate operations not required
treat-dcache-invalidate-as-clean-invalidate	bool	0x0	Treat data cache invalidate operations as clean and invalidate.
undef_ccsidr2_access_for_unimplemented_aarch32	bool	0x0	Whether access to CCSIDR2 is undef if AArch32 is implemented or not
unification-level	int	0x1	Level of Unification Inner Shareable for the cache hierarchy
unification-uniprocessor-level	int	0x1	Level of Unification Uniprocessor for the cache hierarchy

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-81 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
unpred_nested_virtualization_nv_behaviour	int	0x0	Constrained unpredictable choices for HCR_EL2.NV=0 and HCR_EL2.NV1=1 with respect to nested virtualization - 0, Behave as defined in the sepcification as per bit values - 1, Behave as if HCR_EL2.NV=1 and HCR_EL2.NV1=1 for all purpose other than reading back HCR_EL2.NV - 2, Behave as if HCR_EL2.NV=0 and HCR_EL2.NV1=0 for all purpose other than reading back HCR_EL2.NV1
unpred_vncr_el2_ress_mismatch	int	0x0	Constrained unpredictable choices when bits marked as RESS do not all have the same value for VNCR_EL2 - 0, Generating an EL2 translation regime translation abort on use of the VNCR_EL2 register - 1, Reserved sign exteneded bits of VNCR_EL2 are same as bit[52] or bit[48] based on if large VA is supported or not, for all purposes other than reading back the register
unpred_zero_spsr_btype	bool	0x1	Constrained unpredictable control to make SPSR_ELx.BTYPE 0 instead of PSTATE.BTYPE on synchronous exceptions other than Software Step, PC alignment fault, Instruction Abort, Breakpoint or Address Matching Vector Catch, Illegal Execution State, BRK instruction, Branch Target
unpredictable_exclusive_abort_memtype	int	0x0	Cause MMU abort if exclusive access is not supported in certain memory type (0=exclusives allowed in all memory types, 1=exclusives abort in Device memory types, 2=exclusives abort in any type other than WB inner cacheable)
unpredictable_hvc_behaviour	int	0x0	HVC unpredictable behaviour. 0, UNDEF. 1, NOP.
unpredictable_smc_behaviour	int	0x0	SMC unpredictable behaviour. 0, UNDEF. 1, NOP.
use_Xt_as_LDG_STG_input	bool	0x1	Use new MTE Intructions formats for LDG/ST(Z)(2)G, which use Xt as an input
use_mte_eac_02_instructions_encoding	bool	0x1	Use new MTE Intructions encoding since MTE spec EAC 0.2 in ARMv8.5.
use_mte_eac_08_tfsr_encoding	bool	0x1	Use new MTE TFSR_ELx encodings since MTE spec EAC 0.8 in ARMv8.5.
use_rosetta_disass	int	0x1	Use Rosetta disassembly library. Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
use_tlb_contig_hint	bool	0x0	Translation table entries with the contiguous hint bit set generate large TLB entries.
walk_cache_latency	int	0x0	Walk cache latency for TA (Timing Annotation), expressed in simulation ticks
warn_unpredictable_in_v7	bool	0x1	If true, behaviour which is unpredictable in V7 yet is predictable in V8 will produce a warning
watchpoint-log2secondary_restriction	int	0x0	log2 size of secondary restriction of FAR/EDWAR possible values on watchpoint hit for load/store operations.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-81 Parameters for Cluster_ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
wnr_is_read_for_s2f_on_s1_atomic_instr_fault	bool	0x0	Whether WnR is 0 for stage2 fault on stage1 for atomic instructions
wnr_is_read_for_s2f_on_s1_dbm_update	bool	0x0	Whether WnR is 0 for stage2 fault on stage1 descriptor dbm update

Table 3-82 Parameters for ARMAEMv8-A_MP

Name	Type	Default value	Description
cpu0.CFGEND	bool	0x0	Endianness configuration at reset. 0, little endian. 1, big endian.
cpu0.CONFIG64	bool	0x1	Register width configuration at reset. 0, AArch32. 1, AArch64.
cpu0.CP15SDISABLE	bool	0x0	Initialize to disable access to some CP15 registers
cpu0.CP15SDISABLE2	bool	0x0	Initialize to disable access to some CP15 registers
cpu0.CRYPTODISABLE	bool	0x0	Disable cryptographic features.
cpu0.DCZID-log2-block-size	int	0x8	Log2 of the block size cleared by DC ZVA instruction (as read from DCZID_EL0).
cpu0.DCZVA_single_write	bool	0x0	Execute the DCZVA as a single write
cpu0.MPIDR-override	int	0x0	Override of MPIDR value. If nonzero will override the MT, cluster and CPU ID bits in MPIDR.
cpu0.RVBAR	int	0x0	Value of RVBAR_ELx register.
cpu0.RVBAR32	int	0x0	Reset vector address in AARCH32 when VINITHI is not set and ignore_rvbar_in_aarch32 is set
cpu0.SMPnAMP	bool	0x1	Enable broadcast messages necessary for correct SMP operation at reset.
cpu0.TEINIT	bool	0x0	Instruction set state when resetting into AArch32. 0, A32. 1, T32.
cpu0.VINITHI	bool	0x0	Reset value of SCTLR.V.
cpu0.ase-present	bool	0x1	Set whether the model has been built with NEON support
cpu0.clock_divider	int	0x1	Clock divider ratio for asymmetric MP clocking.
cpu0.clock_multiplier	int	0x1	Clock divider ratio for asymmetric MP clocking.
cpu0.crypto_aes	int	0x2	AES instructions supported (requires CryptoPlugin to be loaded). 0, not implemented. 1, AES instructions implemented. 2, AES and PMULL instructions implemented.
cpu0.crypto_sha1	int	0x1	SHA-1 instructions supported (requires CryptoPlugin to be loaded). 0, not implemented. 1, SHA1 instructions implemented.
cpu0.crypto_sha256	int	0x1	SHA-256 instructions supported (requires CryptoPlugin to be loaded). 0, not implemented. 1, SHA256 instructions implemented.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-82 Parameters for ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
cpu0.crypto_sha3	int	0x0	Implement ARMv8.4 SHA-3 instructions (requires CryptoPlugin to be loaded). Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
cpu0.crypto_sha512	int	0x0	Implement ARMv8.4 SHA-512 instructions (requires CryptoPlugin to be loaded). Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
cpu0.crypto_sm3	int	0x0	Implement ARMv8.4 SM-3 instructions (requires CryptoPlugin to be loaded). Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
cpu0.crypto_sm4	int	0x0	Implement ARMv8.4 SM-4 instructions (requires CryptoPlugin to be loaded). Possible values of this parameter are: - 0, feature is not enabled. - 1, feature is implemented if ARMv8.4 is enabled. - 2, feature is implemented.
cpu0.cti-intack_mask	int	0x1	Set bits represent that the corresponding trigger requires software acknowledge via CTIINTACK
cpu0.cti-number_of_claim_bits	int	0x0	Number of implemented bits in CTICLAIMSET
cpu0.cti-number_of_triggers	int	0x8	Number of cti event triggers (default: 8, valid values: {3, 8-32})
cpu0.enable_crc32	int	0x0	CRC32 instructions supported. 0, not implemented. 1, CRC32 instructions implemented.
cpu0.enable_trace_special_hlt_imm16	bool	0x0	Enable usage of parameter trace_special_hlt_imm16
cpu0.etm-present	bool	0x1	Set whether the model has ETM support
cpu0.force-fpsid	bool	0x0	Override the FPSID value
cpu0.force-fpsid-value	int	0x0	Value to override the FPSID value to
cpu0.has_hcptr_tase	bool	0x1	If false, HCPTR.TASE is RES0
cpu0.max_code_cache_mb	int	0x100	Maximum size of the simulation code cache (MiB). For platforms with more than 2 cores this limit will be scaled down. (e.g 1/8 for 16 or more cores)
cpu0.min_sync_level	int	0x0	Force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)
cpu0.number-of-breakpoints	int	0x10	Number of breakpoints.
cpu0.number-of-context-breakpoints	int	0x10	Number of breakpoints that are context aware.
cpu0.number-of-watchpoints	int	0x10	Number of watchpoints.
cpu0.operation_bandwidth	int	0x1	Operation width for ARMv8.4 PMU extension
cpu0.semihosting-A32_HLT	int	0xf000	A32 HLT number for semihosting calls.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-82 Parameters for ARMAEMv8-A_MP (continued)

Name	Type	Default value	Description
cpu0.semihosting-A64_HLT	int	0xf000	A64 HLT number for semihosting calls.
cpu0.semihosting-ARM_SVC	int	0x123456	A32 SVC number for semihosting calls.
cpu0.semihosting-T32_HLT	int	0x3c	T32 HLT number for semihosting calls.
cpu0.semihosting-Thumb_SVC	int	0xab	T32 SVC number for semihosting calls.
cpu0.semihosting-cmd_line	string	""	Command line available to semihosting calls.
cpu0.semihosting-cwd	string	""	Base directory for semihosting file access.
cpu0.semihosting-enable	bool	0x1	Enable semihosting SVC/HLT traps.
cpu0.semihosting-heap_base	int	0x0	Virtual address of heap base.
cpu0.semihosting-heap_limit	int	0xf000000	Virtual address of top of heap.
cpu0.semihosting-prefix	bool	0x0	Prefix semihosting output with target instance name
cpu0.semihosting-stack_base	int	0x10000000	Virtual address of base of descending stack.
cpu0.semihosting-stack_limit	int	0xf000000	Virtual address of stack limit.
cpu0.semihosting-stderr_istty	bool	0x1	Result for semihost istty call when argument is stderr
cpu0.semihosting-stdin_istty	bool	0x1	Result for semihost istty call when argument is stdin
cpu0.semihosting-stdout_istty	bool	0x1	Result for semihost istty call when argument is stdout
cpu0.semihosting-use_stderr	bool	0x0	Send stderr from the simulated process to host stderr
cpu0.trace_special_hlt_imm16	int	0xf000	For this HLT number, IF enable_trace_special_hlt_imm16=true, skip performing usual HLT execution but call MTI trace if registered
cpu0.unpredictable_WPMASKANDBAS	int	0x1	Constrained unpredictable handling of watchpoints when mask and BAS fields specified. 0, IGNOREMASK. 1, IGNOREBAS (default). 2, REPEATBAS8. 3, REPEATBAS.
cpu0.vfp-enable_at_reset	bool	0x0	Enable VFP in CPACR, CPPWR, NSACR at reset. Warning: Arm recommends going through the implementation's suggested VFP power-up sequence!
cpu0.vfp-present	bool	0x1	Set whether the model has VFP support
cpu0.vfp-traps	bool	0x1	Implement support for trapping floating-point exceptions
cpu0.vfp-traps-show-all	bool	0x0	Report all trapped floating-point exceptions in the syndrome when a combination occurs.
cpu0.wfet_early_or_delayed_timeout	int	0x0	WFET early or delayed timeout beyond the threshold value of CNTVCT_EL0 in percentage.
cpu0.wfit_early_or_delayed_timeout	int	0x0	WFIT early or delayed timeout beyond the threshold value of CNTVCT_EL0 in percentage.

3.4.4 ARMAEMv8MCT

ARMAEMv8MCT CPU component. This model is written in C++.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ARMAEMv8MCT contains the following CADI targets:

- ARM_AEMv8M

ARMAEMv8MCT contains the following MTI components:

- *4.5 ARM_AEMv8M - trace* on page 4-946
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402
- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.108 PVCache - trace* on page 4-2761

Table 3-83 Ports

Name	Protocol	Type	Description
ahbd	PVBus	Slave	Debug AHB - core bus slave driven by the DAP.
ahbp_m	PVBus	Master	The core will generate Vendor System data accesses on this port.
ahbs	PVBus	Slave	External master (e.g. DMA) can write TCMs (whether or not enabled in xTCMCR).
auxfault	*Value* on page 2-85	Slave	This is wired to the Auxiliary Fault Status Register.
bigend	*Signal* on page 2-85	Slave	Configure big endian data format.
clk_in	ClockSignal	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the core executes instructions.
coproc_bus	*CoprocBusProtocol* on page 2-81	Slave	Co-Processor Interface
cpuwait	*Signal* on page 2-85	Slave	When this signal is HIGH out of reset, it forces the processor into a quiescent state that delays its boot-up sequence and instruction execution until this signal is driven LOW.
currpri	*Value* on page 2-85	Master	Current execution priority.
dap_s	PVBus	Slave	Debug Access Port (DAP).
dbgen	*Signal* on page 2-85	Slave	Invasive debug enable.
dbgrestart	*Signal* on page 2-85	Slave	External debug request.
dbgrestarted	*Signal* on page 2-85	Master	External debug request.
edbgrq	*Signal* on page 2-85	Slave	External debug request.
etm_reset	*Signal* on page 2-85	Slave	Separate reset for ETM, if param "has_etm_reset" is true.
event	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE and corresponds to the RTL TXEV and RXEV signals.
fpxxc	*Value* on page 2-85	Master	Port which sends the value of the FPXXC cumulative exception flags.
halted	*Signal* on page 2-85	Master	External debug request.
idau	PVBus	Master	The core will generate IDAU Bus request.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-83 Ports (continued)

Name	Protocol	Type	Description
idau_invalidate_region	*Value_64* on page 2-85	Slave	64 bit number to invalid IDAU memory ranage (start_address<<32|end_address)
initpahben	*Signal* on page 2-85	Slave	Enable P-AHB on the next reset
initvtor_ns	*Value* on page 2-85	Slave	Reset configuration port - Non-Secure Vector table offset (VTOR.TBLOFF[31:7]) out of reset This port remains functional no matter ARMv8-M Security Extensions are included or not When ARMv8-M Security Extensions are not included, all exceptions will use NS vector base address given by this port.
initvtor_s	*Value* on page 2-85	Slave	Reset configuration port - Secure Vector table offset (VTOR.TBLOFF[31:7]) out of reset It becomes functional when ARMv8-M Security Extensions are included When ARMv8-M Security Extensions are not included, this port will be ignored.
intisr[496]	*Signal* on page 2-85	Slave	This signal array delivers signals to the NVIC.
intnmi	*Signal* on page 2-85	Slave	Configure non maskable interrupt.
intnum	*Value* on page 2-85	Master	Exception number of the current execution context (from IPSR[8:0]) When the processor is in Thread mode, INTNUM is 0 When the processor is in Handler mode, INTNUM is the exception number of the currently-executing exception.
io_port_in	PVBus	Slave	I/O port pair. See the documentation for the io_port_out port.
io_port_out	PVBus	Master	I/O port pair. Used if IOP is true. Transactions from io_port_out which do not "match" should be returned via io_port_in. For performance reasons, the I/O port interface is not modelled directly. Instead, a simple PVBus gasket is inserted at the point in the memory system where the I/O port would be. In hardware, a device would be attached to the port which would tell the CPU whether it would like to intercept each transaction, given its address. This can be modelled in a performant manner by connecting a PVBusMapper-based device to io_port_out which intercepts transactions of interest and passes other transactions back to the CPU via io_port_in. Your I/O port device model is also responsible for aborting transactions which would be aborted on hardware (e.g. exclusives) if necessary.
lockdcaic	*Signal* on page 2-85	Slave	-
locknsmpu	*Signal* on page 2-85	Slave	Disable writes to the Non-Secure MPU_*_NS registers
locknsvtor	*Signal* on page 2-85	Slave	Disable writes to VTOR_NS
lockpahb	*Signal* on page 2-85	Slave	P-AHB related ports Disable writes to PAHBCR
locksau	*Signal* on page 2-85	Slave	Disable writes to the SAU_* registers
locksmpu	*Signal* on page 2-85	Slave	Disable writes to the Secure MPU_* registers

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-83 Ports (continued)

Name	Protocol	Type	Description
locksvtaircr	*Signal* on page 2-85	Slave	Disable writes to VTOR_S, AIRCR.PRIS, AIRCR.BFHFNMINS
lockup	*Signal* on page 2-85	Master	Asserted when the processor is in lockup state.
niden	*Signal* on page 2-85	Slave	Non-invasive debug enable.
poreset	*Signal* on page 2-85	Slave	Raising this signal will do a power-on reset of the core.
pv_ppbus_m	PVBus	Master	The core will generate External Private Peripheral Bus requests on this port.
pvbus_m	PVBus	Master	The core will generate bus requests on this port.
sleepdeep	*Signal* on page 2-85	Master	Asserted when the processor is in deep sleep.
sleeping	*Signal* on page 2-85	Master	Asserted when the processor is in sleep.
spiden	*Signal* on page 2-85	Slave	Secure invasive debug enable.
spniden	*Signal* on page 2-85	Slave	Secure non-invasive debug enable.
stcalib[2]	*Value* on page 2-85	Slave	This is the calibration value for the SysTick timer.
stclk	ClockSignal	Slave	This is the reference clock for the SysTick timer.
sysreset	*Signal* on page 2-85	Slave	Raising this signal will put the core into reset mode (but does not reset the debug logic).
sysresetreq	*Signal* on page 2-85	Master	Asserted to indicate that a reset is required.
ticks	*InstructionCount* on page 2-84	Master	Port allowing the number of instructions since startup to be read from the CPU.

Table 3-84 Parameters for ARM_AEMv8M

Name	Type	Default value	Description
AIRCR.ENDIANNESS	bool	0x0	Initialize processor to big endian mode
BB_PRESENT	bool	0x0	Enable bitbanding
CFGMEMALIAS	int	0x0	Memory address alias bit for the ITCM, DTCM and P-AHB regions. 0=No alias, 1=Alias bit 24, 2=Alias bit 25, 4=Alias bit 26, 8=Alias bit 27, 16=Alias bit 28
CFGPAHBSZ	int	0x0	Size of the P-AHB peripheral port memory region. 0=P-AHB disabled, 1=64MB, 2=128MB, 3=256MB, 4=512MB
DTGU	bool	0x0	DTCM Security Gate Unit included
DTGUBLKSZ	int	0x3	DTCM gate unit block size. Size=pow(2, DTGUBLKSZ + 5) bytes

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-84 Parameters for ARM_AEMv8M (continued)

Name	Type	Default value	Description
DTGUMAXBLKS	int	0x0	Maxiumum number of DTCM gate unit blocks. Number of blocks=pow(2, DTGUMAXBLKS)
DWT_CTRL.NOCYCCNT	bool	0x0	DWT cycle-counter not present (v8M_bl/v6M never have one).
DWT_CTRL.NOPRFCNT	bool	0x0	DWT performance-counters not present (v8M_bl/v6M never have them).
DWT_CTRL.NUMCOMP	int	0x4	Number of watchpoint unit comparators implemented
DWT_DEVARCH.REVISION	int	0x1	0: V2, 1: V2.1.
DWT_FUNCTION0.ID	int	0xb	Sets the capabilities of the comparator that is accessible via the register, DWT_FUNCTION0. If 'baseline' is set, invalid ID bits are cleared
DWT_FUNCTION1.ID	int	0x1e	Sets the capabilities of the comparator that is accessible via the register, DWT_FUNCTION1. If 'baseline' is set, invalid ID bits are cleared
DWT_FUNCTION10.ID	int	0xb	Sets the capabilities of the comparator that is accessible via the register, DWT_FUNCTION10. If 'baseline' is set, invalid ID bits are cleared
DWT_FUNCTION11.ID	int	0x1e	Sets the capabilities of the comparator that is accessible via the register, DWT_FUNCTION11. If 'baseline' is set, invalid ID bits are cleared
DWT_FUNCTION12.ID	int	0xb	Sets the capabilities of the comparator that is accessible via the register, DWT_FUNCTION12. If 'baseline' is set, invalid ID bits are cleared
DWT_FUNCTION13.ID	int	0x1e	Sets the capabilities of the comparator that is accessible via the register, DWT_FUNCTION13. If 'baseline' is set, invalid ID bits are cleared
DWT_FUNCTION14.ID	int	0xb	Sets the capabilities of the comparator that is accessible via the register, DWT_FUNCTION14. If 'baseline' is set, invalid ID bits are cleared

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-84 Parameters for ARM_AEMv8M (continued)

Name	Type	Default value	Description
DWT_FUNCTION15.ID	int	0x1e	Sets the capabilities of the comparator that is accessible via the register, DWT_FUNCTION15. If 'baseline' is set, invalid ID bits are cleared
DWT_FUNCTION2.ID	int	0xb	Sets the capabilities of the comparator that is accessible via the register, DWT_FUNCTION2. If 'baseline' is set, invalid ID bits are cleared
DWT_FUNCTION3.ID	int	0x1e	Sets the capabilities of the comparator that is accessible via the register, DWT_FUNCTION3. If 'baseline' is set, invalid ID bits are cleared
DWT_FUNCTION4.ID	int	0xb	Sets the capabilities of the comparator that is accessible via the register, DWT_FUNCTION4. If 'baseline' is set, invalid ID bits are cleared
DWT_FUNCTION5.ID	int	0x1e	Sets the capabilities of the comparator that is accessible via the register, DWT_FUNCTION5. If 'baseline' is set, invalid ID bits are cleared
DWT_FUNCTION6.ID	int	0xb	Sets the capabilities of the comparator that is accessible via the register, DWT_FUNCTION6. If 'baseline' is set, invalid ID bits are cleared
DWT_FUNCTION7.ID	int	0x1e	Sets the capabilities of the comparator that is accessible via the register, DWT_FUNCTION7. If 'baseline' is set, invalid ID bits are cleared
DWT_FUNCTION8.ID	int	0xb	Sets the capabilities of the comparator that is accessible via the register, DWT_FUNCTION8. If 'baseline' is set, invalid ID bits are cleared
DWT_FUNCTION9.ID	int	0x1e	Sets the capabilities of the comparator that is accessible via the register, DWT_FUNCTION9. If 'baseline' is set, invalid ID bits are cleared

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-84 Parameters for ARM_AEMv8M (continued)

Name	Type	Default value	Description
DWT_TRACE	bool	0x1	Support for DWT trace, controls the DWT_CTRL.NOTRCPKT bit. false : No DWT trace included, true: DWT trace included
FP_CTRL.NUM_CODE	int	0x8	Number of breakpoint unit comparators implemented (limited to 15 in V6M or baseline)
FP_CTRL.NUM_LIT	int	0x0	How many Literals FPB supports remapping (ignored if baseline or TZM)
FP_REMAP.RMPSPT	bool	0x1	FPB supports remapping (ignored if baseline or SECEXT)
ID_DFR0.Debug_Model_M_profile	bool	0x1	Set whether debug extensions are implemented
INITPAHBEN	bool	0x0	The P-AHB enable state at reset
INITVTOR_NS	int	0x0	Non-Secure vector-table offset at reset
INITVTOR_S	int	0x0	Secure vector-table offset at reset
ITGU	bool	0x0	ITCM Security Gate Unit included
ITGUBLKSZ	int	0x3	ITCM gate unit block size. Size=pow(2, ITGUBLKSZ + 5) bytes
ITGUMAXBLKS	int	0x0	Maxiumum number of ITCM gate unit blocks. Number of blocks=pow(2, ITGUMAXBLKS)
ITM	bool	0x1	Level of instrumentation trace supported. false : No ITM trace included, true: ITM trace included (unless baseline)
ITM_HAS_LSR	bool	0x1	ITM support LAR and LSR for software lock
LOCKDTGU	bool	0x0	Lock down of Data TGU registers write
LOCKITGU	bool	0x0	Lock down of Instruction TGU registers write
LOCKTCM	bool	0x0	Lock down of TCM registers write
LVL_WIDTH	int	0x3	Number of bits of interrupt priority (baseline has 2)
MPU_TYPE_NS.DREGION	int	0x10	Number of regions in the Non-Secure MPU. If Security Extentions are absent, this is the total number of MPU regions

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-84 Parameters for ARM_AEMv8M (continued)

Name	Type	Default value	Description
MPU_TYPE_S.DREGION	int	0x10	Number of regions in the Secure MPU. If Security Extentions are absent, this is ignored
MVFR0.Double-precision	bool	0x1	Support 8-byte floats
NUM_IRQ	int	0x10	Number of user interrupts
REGISTER_POP_ORDER	string	"R0-R3,R12,R14,RETURN_ADDR,CPSR,S0-S15,FPSCR,PADDING,S16-S31"	Order in which the registers are popped off the stack during exception return. A comma separated list of register names and ranges.
REGISTER_PUSH_ORDER	string	"R0-R3,R12,R14,RETURN_ADDR,CPSR,S0-S15,FPSCR,PADDING,S16-S31"	Order in which the registers are pushed on to the stack during exception handling. A comma separated list of register names and ranges.
SAU_CTRL.ALLNS	bool	0x0	At reset, the SAU treats entire memory space as NS when the SAU is disabled if this is set
SAU_CTRL.ENABLE	bool	0x0	Enable SAU at reset
SAU_REGION0.BADDR	int	0x0	Base address of SAU region0 at reset
SAU_REGION0.ENABLE	bool	0x0	Enable SAU region0 at reset
SAU_REGION0.LADDR	int	0x0	Limit address of SAU region0 at reset
SAU_REGION0.NSC	bool	0x0	Set NSC for SAU region0 at reset
SAU_REGION1.BADDR	int	0x0	Base address of SAU region1 at reset
SAU_REGION1.ENABLE	bool	0x0	Enable SAU region1 at reset
SAU_REGION1.LADDR	int	0x0	Limit address of SAU region1 at reset
SAU_REGION1.NSC	bool	0x0	Set NSC for SAU region1 at reset
SAU_REGION10.BADDR	int	0x0	Base address of SAU region10 at reset
SAU_REGION10.ENABLE	bool	0x0	Enable SAU region10 at reset
SAU_REGION10.LADDR	int	0x0	Limit address of SAU region10 at reset
SAU_REGION10.NSC	bool	0x0	Set NSC for SAU region10 at reset
SAU_REGION11.BADDR	int	0x0	Base address of SAU region11 at reset
SAU_REGION11.ENABLE	bool	0x0	Enable SAU region11 at reset
SAU_REGION11.LADDR	int	0x0	Limit address of SAU region11 at reset
SAU_REGION11.NSC	bool	0x0	Set NSC for SAU region11 at reset

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-84 Parameters for ARM_AEMv8M (continued)

Name	Type	Default value	Description
SAU_REGION12.BADDR	int	0x0	Base address of SAU region12 at reset
SAU_REGION12.ENABLE	bool	0x0	Enable SAU region12 at reset
SAU_REGION12.LADDR	int	0x0	Limit address of SAU region12 at reset
SAU_REGION12.NSC	bool	0x0	Set NSC for SAU region12 at reset
SAU_REGION13.BADDR	int	0x0	Base address of SAU region13 at reset
SAU_REGION13.ENABLE	bool	0x0	Enable SAU region13 at reset
SAU_REGION13.LADDR	int	0x0	Limit address of SAU region13 at reset
SAU_REGION13.NSC	bool	0x0	Set NSC for SAU region13 at reset
SAU_REGION14.BADDR	int	0x0	Base address of SAU region14 at reset
SAU_REGION14.ENABLE	bool	0x0	Enable SAU region14 at reset
SAU_REGION14.LADDR	int	0x0	Limit address of SAU region14 at reset
SAU_REGION14.NSC	bool	0x0	Set NSC for SAU region14 at reset
SAU_REGION15.BADDR	int	0x0	Base address of SAU region15 at reset
SAU_REGION15.ENABLE	bool	0x0	Enable SAU region15 at reset
SAU_REGION15.LADDR	int	0x0	Limit address of SAU region15 at reset
SAU_REGION15.NSC	bool	0x0	Set NSC for SAU region15 at reset
SAU_REGION16.BADDR	int	0x0	Base address of SAU region16 at reset
SAU_REGION16.ENABLE	bool	0x0	Enable SAU region16 at reset
SAU_REGION16.LADDR	int	0x0	Limit address of SAU region16 at reset
SAU_REGION16.NSC	bool	0x0	Set NSC for SAU region16 at reset
SAU_REGION17.BADDR	int	0x0	Base address of SAU region17 at reset
SAU_REGION17.ENABLE	bool	0x0	Enable SAU region17 at reset
SAU_REGION17.LADDR	int	0x0	Limit address of SAU region17 at reset
SAU_REGION17.NSC	bool	0x0	Set NSC for SAU region17 at reset
SAU_REGION18.BADDR	int	0x0	Base address of SAU region18 at reset
SAU_REGION18.ENABLE	bool	0x0	Enable SAU region18 at reset

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-84 Parameters for ARM_AEMv8M (continued)

Name	Type	Default value	Description
SAU_REGION18.LADDR	int	0x0	Limit address of SAU region18 at reset
SAU_REGION18.NSC	bool	0x0	Set NSC for SAU region18 at reset
SAU_REGION19.BADDR	int	0x0	Base address of SAU region19 at reset
SAU_REGION19.ENABLE	bool	0x0	Enable SAU region19 at reset
SAU_REGION19.LADDR	int	0x0	Limit address of SAU region19 at reset
SAU_REGION19.NSC	bool	0x0	Set NSC for SAU region19 at reset
SAU_REGION2.BADDR	int	0x0	Base address of SAU region2 at reset
SAU_REGION2.ENABLE	bool	0x0	Enable SAU region2 at reset
SAU_REGION2.LADDR	int	0x0	Limit address of SAU region2 at reset
SAU_REGION2.NSC	bool	0x0	Set NSC for SAU region2 at reset
SAU_REGION20.BADDR	int	0x0	Base address of SAU region20 at reset
SAU_REGION20.ENABLE	bool	0x0	Enable SAU region20 at reset
SAU_REGION20.LADDR	int	0x0	Limit address of SAU region20 at reset
SAU_REGION20.NSC	bool	0x0	Set NSC for SAU region20 at reset
SAU_REGION21.BADDR	int	0x0	Base address of SAU region21 at reset
SAU_REGION21.ENABLE	bool	0x0	Enable SAU region21 at reset
SAU_REGION21.LADDR	int	0x0	Limit address of SAU region21 at reset
SAU_REGION21.NSC	bool	0x0	Set NSC for SAU region21 at reset
SAU_REGION22.BADDR	int	0x0	Base address of SAU region22 at reset
SAU_REGION22.ENABLE	bool	0x0	Enable SAU region22 at reset
SAU_REGION22.LADDR	int	0x0	Limit address of SAU region22 at reset
SAU_REGION22.NSC	bool	0x0	Set NSC for SAU region22 at reset
SAU_REGION23.BADDR	int	0x0	Base address of SAU region23 at reset
SAU_REGION23.ENABLE	bool	0x0	Enable SAU region23 at reset
SAU_REGION23.LADDR	int	0x0	Limit address of SAU region23 at reset
SAU_REGION23.NSC	bool	0x0	Set NSC for SAU region23 at reset

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-84 Parameters for ARM_AEMv8M (continued)

Name	Type	Default value	Description
SAU_REGION24.BADDR	int	0x0	Base address of SAU region24 at reset
SAU_REGION24.ENABLE	bool	0x0	Enable SAU region24 at reset
SAU_REGION24.LADDR	int	0x0	Limit address of SAU region24 at reset
SAU_REGION24.NSC	bool	0x0	Set NSC for SAU region24 at reset
SAU_REGION25.BADDR	int	0x0	Base address of SAU region25 at reset
SAU_REGION25.ENABLE	bool	0x0	Enable SAU region25 at reset
SAU_REGION25.LADDR	int	0x0	Limit address of SAU region25 at reset
SAU_REGION25.NSC	bool	0x0	Set NSC for SAU region25 at reset
SAU_REGION26.BADDR	int	0x0	Base address of SAU region26 at reset
SAU_REGION26.ENABLE	bool	0x0	Enable SAU region26 at reset
SAU_REGION26.LADDR	int	0x0	Limit address of SAU region26 at reset
SAU_REGION26.NSC	bool	0x0	Set NSC for SAU region26 at reset
SAU_REGION27.BADDR	int	0x0	Base address of SAU region27 at reset
SAU_REGION27.ENABLE	bool	0x0	Enable SAU region27 at reset
SAU_REGION27.LADDR	int	0x0	Limit address of SAU region27 at reset
SAU_REGION27.NSC	bool	0x0	Set NSC for SAU region27 at reset
SAU_REGION28.BADDR	int	0x0	Base address of SAU region28 at reset
SAU_REGION28.ENABLE	bool	0x0	Enable SAU region28 at reset
SAU_REGION28.LADDR	int	0x0	Limit address of SAU region28 at reset
SAU_REGION28.NSC	bool	0x0	Set NSC for SAU region28 at reset
SAU_REGION29.BADDR	int	0x0	Base address of SAU region29 at reset
SAU_REGION29.ENABLE	bool	0x0	Enable SAU region29 at reset
SAU_REGION29.LADDR	int	0x0	Limit address of SAU region29 at reset
SAU_REGION29.NSC	bool	0x0	Set NSC for SAU region29 at reset
SAU_REGION3.BADDR	int	0x0	Base address of SAU region3 at reset
SAU_REGION3.ENABLE	bool	0x0	Enable SAU region3 at reset

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-84 Parameters for ARM_AEMv8M (continued)

Name	Type	Default value	Description
SAU_REGION3.LADDR	int	0x0	Limit address of SAU region3 at reset
SAU_REGION3.NSC	bool	0x0	Set NSC for SAU region3 at reset
SAU_REGION30.BADDR	int	0x0	Base address of SAU region30 at reset
SAU_REGION30.ENABLE	bool	0x0	Enable SAU region30 at reset
SAU_REGION30.LADDR	int	0x0	Limit address of SAU region30 at reset
SAU_REGION30.NSC	bool	0x0	Set NSC for SAU region30 at reset
SAU_REGION31.BADDR	int	0x0	Base address of SAU region31 at reset
SAU_REGION31.ENABLE	bool	0x0	Enable SAU region31 at reset
SAU_REGION31.LADDR	int	0x0	Limit address of SAU region31 at reset
SAU_REGION31.NSC	bool	0x0	Set NSC for SAU region31 at reset
SAU_REGION32.BADDR	int	0x0	Base address of SAU region32 at reset
SAU_REGION32.ENABLE	bool	0x0	Enable SAU region32 at reset
SAU_REGION32.LADDR	int	0x0	Limit address of SAU region32 at reset
SAU_REGION32.NSC	bool	0x0	Set NSC for SAU region32 at reset
SAU_REGION33.BADDR	int	0x0	Base address of SAU region33 at reset
SAU_REGION33.ENABLE	bool	0x0	Enable SAU region33 at reset
SAU_REGION33.LADDR	int	0x0	Limit address of SAU region33 at reset
SAU_REGION33.NSC	bool	0x0	Set NSC for SAU region33 at reset
SAU_REGION34.BADDR	int	0x0	Base address of SAU region34 at reset
SAU_REGION34.ENABLE	bool	0x0	Enable SAU region34 at reset
SAU_REGION34.LADDR	int	0x0	Limit address of SAU region34 at reset
SAU_REGION34.NSC	bool	0x0	Set NSC for SAU region34 at reset
SAU_REGION35.BADDR	int	0x0	Base address of SAU region35 at reset
SAU_REGION35.ENABLE	bool	0x0	Enable SAU region35 at reset
SAU_REGION35.LADDR	int	0x0	Limit address of SAU region35 at reset
SAU_REGION35.NSC	bool	0x0	Set NSC for SAU region35 at reset

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-84 Parameters for ARM_AEMv8M (continued)

Name	Type	Default value	Description
SAU_REGION36.BADDR	int	0x0	Base address of SAU region36 at reset
SAU_REGION36.ENABLE	bool	0x0	Enable SAU region36 at reset
SAU_REGION36.LADDR	int	0x0	Limit address of SAU region36 at reset
SAU_REGION36.NSC	bool	0x0	Set NSC for SAU region36 at reset
SAU_REGION37.BADDR	int	0x0	Base address of SAU region37 at reset
SAU_REGION37.ENABLE	bool	0x0	Enable SAU region37 at reset
SAU_REGION37.LADDR	int	0x0	Limit address of SAU region37 at reset
SAU_REGION37.NSC	bool	0x0	Set NSC for SAU region37 at reset
SAU_REGION38.BADDR	int	0x0	Base address of SAU region38 at reset
SAU_REGION38.ENABLE	bool	0x0	Enable SAU region38 at reset
SAU_REGION38.LADDR	int	0x0	Limit address of SAU region38 at reset
SAU_REGION38.NSC	bool	0x0	Set NSC for SAU region38 at reset
SAU_REGION39.BADDR	int	0x0	Base address of SAU region39 at reset
SAU_REGION39.ENABLE	bool	0x0	Enable SAU region39 at reset
SAU_REGION39.LADDR	int	0x0	Limit address of SAU region39 at reset
SAU_REGION39.NSC	bool	0x0	Set NSC for SAU region39 at reset
SAU_REGION4.BADDR	int	0x0	Base address of SAU region4 at reset
SAU_REGION4.ENABLE	bool	0x0	Enable SAU region4 at reset
SAU_REGION4.LADDR	int	0x0	Limit address of SAU region4 at reset
SAU_REGION4.NSC	bool	0x0	Set NSC for SAU region4 at reset
SAU_REGION40.BADDR	int	0x0	Base address of SAU region40 at reset
SAU_REGION40.ENABLE	bool	0x0	Enable SAU region40 at reset
SAU_REGION40.LADDR	int	0x0	Limit address of SAU region40 at reset
SAU_REGION40.NSC	bool	0x0	Set NSC for SAU region40 at reset
SAU_REGION41.BADDR	int	0x0	Base address of SAU region41 at reset
SAU_REGION41.ENABLE	bool	0x0	Enable SAU region41 at reset

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-84 Parameters for ARM_AEMv8M (continued)

Name	Type	Default value	Description
SAU_REGION41.LADDR	int	0x0	Limit address of SAU region41 at reset
SAU_REGION41.NSC	bool	0x0	Set NSC for SAU region41 at reset
SAU_REGION42.BADDR	int	0x0	Base address of SAU region42 at reset
SAU_REGION42.ENABLE	bool	0x0	Enable SAU region42 at reset
SAU_REGION42.LADDR	int	0x0	Limit address of SAU region42 at reset
SAU_REGION42.NSC	bool	0x0	Set NSC for SAU region42 at reset
SAU_REGION43.BADDR	int	0x0	Base address of SAU region43 at reset
SAU_REGION43.ENABLE	bool	0x0	Enable SAU region43 at reset
SAU_REGION43.LADDR	int	0x0	Limit address of SAU region43 at reset
SAU_REGION43.NSC	bool	0x0	Set NSC for SAU region43 at reset
SAU_REGION44.BADDR	int	0x0	Base address of SAU region44 at reset
SAU_REGION44.ENABLE	bool	0x0	Enable SAU region44 at reset
SAU_REGION44.LADDR	int	0x0	Limit address of SAU region44 at reset
SAU_REGION44.NSC	bool	0x0	Set NSC for SAU region44 at reset
SAU_REGION45.BADDR	int	0x0	Base address of SAU region45 at reset
SAU_REGION45.ENABLE	bool	0x0	Enable SAU region45 at reset
SAU_REGION45.LADDR	int	0x0	Limit address of SAU region45 at reset
SAU_REGION45.NSC	bool	0x0	Set NSC for SAU region45 at reset
SAU_REGION46.BADDR	int	0x0	Base address of SAU region46 at reset
SAU_REGION46.ENABLE	bool	0x0	Enable SAU region46 at reset
SAU_REGION46.LADDR	int	0x0	Limit address of SAU region46 at reset
SAU_REGION46.NSC	bool	0x0	Set NSC for SAU region46 at reset
SAU_REGION47.BADDR	int	0x0	Base address of SAU region47 at reset
SAU_REGION47.ENABLE	bool	0x0	Enable SAU region47 at reset
SAU_REGION47.LADDR	int	0x0	Limit address of SAU region47 at reset
SAU_REGION47.NSC	bool	0x0	Set NSC for SAU region47 at reset

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-84 Parameters for ARM_AEMv8M (continued)

Name	Type	Default value	Description
SAU_REGION48.BADDR	int	0x0	Base address of SAU region48 at reset
SAU_REGION48.ENABLE	bool	0x0	Enable SAU region48 at reset
SAU_REGION48.LADDR	int	0x0	Limit address of SAU region48 at reset
SAU_REGION48.NSC	bool	0x0	Set NSC for SAU region48 at reset
SAU_REGION49.BADDR	int	0x0	Base address of SAU region49 at reset
SAU_REGION49.ENABLE	bool	0x0	Enable SAU region49 at reset
SAU_REGION49.LADDR	int	0x0	Limit address of SAU region49 at reset
SAU_REGION49.NSC	bool	0x0	Set NSC for SAU region49 at reset
SAU_REGION5.BADDR	int	0x0	Base address of SAU region5 at reset
SAU_REGION5.ENABLE	bool	0x0	Enable SAU region5 at reset
SAU_REGION5.LADDR	int	0x0	Limit address of SAU region5 at reset
SAU_REGION5.NSC	bool	0x0	Set NSC for SAU region5 at reset
SAU_REGION50.BADDR	int	0x0	Base address of SAU region50 at reset
SAU_REGION50.ENABLE	bool	0x0	Enable SAU region50 at reset
SAU_REGION50.LADDR	int	0x0	Limit address of SAU region50 at reset
SAU_REGION50.NSC	bool	0x0	Set NSC for SAU region50 at reset
SAU_REGION51.BADDR	int	0x0	Base address of SAU region51 at reset
SAU_REGION51.ENABLE	bool	0x0	Enable SAU region51 at reset
SAU_REGION51.LADDR	int	0x0	Limit address of SAU region51 at reset
SAU_REGION51.NSC	bool	0x0	Set NSC for SAU region51 at reset
SAU_REGION52.BADDR	int	0x0	Base address of SAU region52 at reset
SAU_REGION52.ENABLE	bool	0x0	Enable SAU region52 at reset
SAU_REGION52.LADDR	int	0x0	Limit address of SAU region52 at reset
SAU_REGION52.NSC	bool	0x0	Set NSC for SAU region52 at reset
SAU_REGION53.BADDR	int	0x0	Base address of SAU region53 at reset
SAU_REGION53.ENABLE	bool	0x0	Enable SAU region53 at reset

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-84 Parameters for ARM_AEMv8M (continued)

Name	Type	Default value	Description
SAU_REGION53.LADDR	int	0x0	Limit address of SAU region53 at reset
SAU_REGION53.NSC	bool	0x0	Set NSC for SAU region53 at reset
SAU_REGION54.BADDR	int	0x0	Base address of SAU region54 at reset
SAU_REGION54.ENABLE	bool	0x0	Enable SAU region54 at reset
SAU_REGION54.LADDR	int	0x0	Limit address of SAU region54 at reset
SAU_REGION54.NSC	bool	0x0	Set NSC for SAU region54 at reset
SAU_REGION55.BADDR	int	0x0	Base address of SAU region55 at reset
SAU_REGION55.ENABLE	bool	0x0	Enable SAU region55 at reset
SAU_REGION55.LADDR	int	0x0	Limit address of SAU region55 at reset
SAU_REGION55.NSC	bool	0x0	Set NSC for SAU region55 at reset
SAU_REGION56.BADDR	int	0x0	Base address of SAU region56 at reset
SAU_REGION56.ENABLE	bool	0x0	Enable SAU region56 at reset
SAU_REGION56.LADDR	int	0x0	Limit address of SAU region56 at reset
SAU_REGION56.NSC	bool	0x0	Set NSC for SAU region56 at reset
SAU_REGION57.BADDR	int	0x0	Base address of SAU region57 at reset
SAU_REGION57.ENABLE	bool	0x0	Enable SAU region57 at reset
SAU_REGION57.LADDR	int	0x0	Limit address of SAU region57 at reset
SAU_REGION57.NSC	bool	0x0	Set NSC for SAU region57 at reset
SAU_REGION58.BADDR	int	0x0	Base address of SAU region58 at reset
SAU_REGION58.ENABLE	bool	0x0	Enable SAU region58 at reset
SAU_REGION58.LADDR	int	0x0	Limit address of SAU region58 at reset
SAU_REGION58.NSC	bool	0x0	Set NSC for SAU region58 at reset
SAU_REGION59.BADDR	int	0x0	Base address of SAU region59 at reset
SAU_REGION59.ENABLE	bool	0x0	Enable SAU region59 at reset
SAU_REGION59.LADDR	int	0x0	Limit address of SAU region59 at reset
SAU_REGION59.NSC	bool	0x0	Set NSC for SAU region59 at reset

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-84 Parameters for ARM_AEMv8M (continued)

Name	Type	Default value	Description
SAU_REGION6.BADDR	int	0x0	Base address of SAU region6 at reset
SAU_REGION6.ENABLE	bool	0x0	Enable SAU region6 at reset
SAU_REGION6.LADDR	int	0x0	Limit address of SAU region6 at reset
SAU_REGION6.NSC	bool	0x0	Set NSC for SAU region6 at reset
SAU_REGION60.BADDR	int	0x0	Base address of SAU region60 at reset
SAU_REGION60.ENABLE	bool	0x0	Enable SAU region60 at reset
SAU_REGION60.LADDR	int	0x0	Limit address of SAU region60 at reset
SAU_REGION60.NSC	bool	0x0	Set NSC for SAU region60 at reset
SAU_REGION61.BADDR	int	0x0	Base address of SAU region61 at reset
SAU_REGION61.ENABLE	bool	0x0	Enable SAU region61 at reset
SAU_REGION61.LADDR	int	0x0	Limit address of SAU region61 at reset
SAU_REGION61.NSC	bool	0x0	Set NSC for SAU region61 at reset
SAU_REGION62.BADDR	int	0x0	Base address of SAU region62 at reset
SAU_REGION62.ENABLE	bool	0x0	Enable SAU region62 at reset
SAU_REGION62.LADDR	int	0x0	Limit address of SAU region62 at reset
SAU_REGION62.NSC	bool	0x0	Set NSC for SAU region62 at reset
SAU_REGION63.BADDR	int	0x0	Base address of SAU region63 at reset
SAU_REGION63.ENABLE	bool	0x0	Enable SAU region63 at reset
SAU_REGION63.LADDR	int	0x0	Limit address of SAU region63 at reset
SAU_REGION63.NSC	bool	0x0	Set NSC for SAU region63 at reset
SAU_REGION64.BADDR	int	0x0	Base address of SAU region64 at reset
SAU_REGION64.ENABLE	bool	0x0	Enable SAU region64 at reset
SAU_REGION64.LADDR	int	0x0	Limit address of SAU region64 at reset
SAU_REGION64.NSC	bool	0x0	Set NSC for SAU region64 at reset
SAU_REGION65.BADDR	int	0x0	Base address of SAU region65 at reset
SAU_REGION65.ENABLE	bool	0x0	Enable SAU region65 at reset

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-84 Parameters for ARM_AEMv8M (continued)

Name	Type	Default value	Description
SAU_REGION65.LADDR	int	0x0	Limit address of SAU region65 at reset
SAU_REGION65.NSC	bool	0x0	Set NSC for SAU region65 at reset
SAU_REGION66.BADDR	int	0x0	Base address of SAU region66 at reset
SAU_REGION66.ENABLE	bool	0x0	Enable SAU region66 at reset
SAU_REGION66.LADDR	int	0x0	Limit address of SAU region66 at reset
SAU_REGION66.NSC	bool	0x0	Set NSC for SAU region66 at reset
SAU_REGION67.BADDR	int	0x0	Base address of SAU region67 at reset
SAU_REGION67.ENABLE	bool	0x0	Enable SAU region67 at reset
SAU_REGION67.LADDR	int	0x0	Limit address of SAU region67 at reset
SAU_REGION67.NSC	bool	0x0	Set NSC for SAU region67 at reset
SAU_REGION68.BADDR	int	0x0	Base address of SAU region68 at reset
SAU_REGION68.ENABLE	bool	0x0	Enable SAU region68 at reset
SAU_REGION68.LADDR	int	0x0	Limit address of SAU region68 at reset
SAU_REGION68.NSC	bool	0x0	Set NSC for SAU region68 at reset
SAU_REGION69.BADDR	int	0x0	Base address of SAU region69 at reset
SAU_REGION69.ENABLE	bool	0x0	Enable SAU region69 at reset
SAU_REGION69.LADDR	int	0x0	Limit address of SAU region69 at reset
SAU_REGION69.NSC	bool	0x0	Set NSC for SAU region69 at reset
SAU_REGION7.BADDR	int	0x0	Base address of SAU region7 at reset
SAU_REGION7.ENABLE	bool	0x0	Enable SAU region7 at reset
SAU_REGION7.LADDR	int	0x0	Limit address of SAU region7 at reset
SAU_REGION7.NSC	bool	0x0	Set NSC for SAU region7 at reset
SAU_REGION70.BADDR	int	0x0	Base address of SAU region70 at reset
SAU_REGION70.ENABLE	bool	0x0	Enable SAU region70 at reset
SAU_REGION70.LADDR	int	0x0	Limit address of SAU region70 at reset
SAU_REGION70.NSC	bool	0x0	Set NSC for SAU region70 at reset

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-84 Parameters for ARM_AEMv8M (continued)

Name	Type	Default value	Description
SAU_REGION71.BADDR	int	0x0	Base address of SAU region71 at reset
SAU_REGION71.ENABLE	bool	0x0	Enable SAU region71 at reset
SAU_REGION71.LADDR	int	0x0	Limit address of SAU region71 at reset
SAU_REGION71.NSC	bool	0x0	Set NSC for SAU region71 at reset
SAU_REGION72.BADDR	int	0x0	Base address of SAU region72 at reset
SAU_REGION72.ENABLE	bool	0x0	Enable SAU region72 at reset
SAU_REGION72.LADDR	int	0x0	Limit address of SAU region72 at reset
SAU_REGION72.NSC	bool	0x0	Set NSC for SAU region72 at reset
SAU_REGION73.BADDR	int	0x0	Base address of SAU region73 at reset
SAU_REGION73.ENABLE	bool	0x0	Enable SAU region73 at reset
SAU_REGION73.NSC	bool	0x0	Set NSC for SAU region73 at reset
SAU_REGION8.BADDR	int	0x0	Base address of SAU region8 at reset
SAU_REGION8.ENABLE	bool	0x0	Enable SAU region8 at reset
SAU_REGION8.LADDR	int	0x0	Limit address of SAU region8 at reset
SAU_REGION8.NSC	bool	0x0	Set NSC for SAU region8 at reset
SAU_REGION9.BADDR	int	0x0	Base address of SAU region9 at reset
SAU_REGION9.ENABLE	bool	0x0	Enable SAU region9 at reset
SAU_REGION9.LADDR	int	0x0	Limit address of SAU region9 at reset
SAU_REGION9.NSC	bool	0x0	Set NSC for SAU region9 at reset
SAU_TYPE.SREGION	int	0x10	Number of SAU regions (0 => no SAU)
VTOR_NS	bool	0x1	NonSecure Vector Table Offset Register is writeable
VTOR_NS_MASK	int	0xffffff80	Non-Secure VTOR write mask
VTOR_S	bool	0x1	Secure Vector Table Offset Register is writeable
VTOR_S_MASK	int	0xffffff80	Secure VTOR write mask
allow_debug_monitor_with_in_flight_inst	bool	0x0	Allow handling Debug Monitor exception with in-flight instructions
allow_stack_accesses_to_ppb_space	bool	0x0	Allow stack accesses to PPB space

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-84 Parameters for ARM_AEMv8M (continued)

Name	Type	Default value	Description
callee_register_push_low_to_high	bool	0x1	If true, push callee registers in order from R4 to R11. If false, push R11 to R4
cde_impl_name	string	""	Name of the CDE implementation for this core (implementation contributed by MTI plugin).
cpi_div	int	0x1	divider for calculating CPI (Cycles Per Instruction)
cpi_mul	int	0x1	multiplier for calculating CPI (Cycles Per Instruction)
dcache-invalidate-ns-cleans-s	bool	0x0	Whether V8M DCI* in non-secure should clean-and-invalidate secure cache contents.
dcache-size	int	0x8000	L1 D-cache size in bytes
dcache-state_modelled	bool	0x0	Set whether D-cache has stateful implementation
dcache-ways	int	0x4	L1 D-cache ways (sets are implicit from size)
delay_faultmask_update	bool	0x0	Delay FAULTMASK update to context sync
delay_sysreg_update	bool	0x0	Delay some system register updates (e.g. SHCSR) to context sync
dtcm_enable	bool	0x0	Enable DTCM at reset
dtcm_size	int	0x100	DTCM size in KB
duplicate_CADI_TCM_writes	bool	0x0	CADI writes to TCMs are also sent to downstream memory at same addresses (for validation platforms)
dwt_unaligned_word_access_as_half_word	bool	0x1	DWT Treat unaligned word access as half word or bytes.
exercise_strex_fail	bool	0x0	Reject a pseudo-random majority of exclusive store instructions
has_cde	bool	0x0	Enables Custom Datapath Extensions
has_global_monitor	bool	0x0	Enable global monitor, with granule set to 16 words / 64 bytes
has_writebuffer	bool	0x0	Implement write accesses buffering before L1 cache. May affect ext_abort behaviour.
icache-size	int	0x8000	L1 I-cache size in bytes
icache-state_modelled	bool	0x0	Set whether I-cache has stateful implementation
icache-ways	int	0x2	L1 I-cache ways (sets are implicit from size)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-84 Parameters for ARM_AEMv8M (continued)

Name	Type	Default value	Description
ignore_RNR_top_nibble	bool	0x0	If set, only the bottom four bits of MPU_RNR.REGION are used
ignore_out_of_range_RNR_write	bool	0x0	If an MPU_RNR.REGION write is out of range, ignore it ; if false, MPU_RNR values wrap
itcm_enable	bool	0x0	Enable ITCM at reset
itcm_size	int	0x100	ITCM size in KB
master_id	int	0x0	Master ID presented in bus transactions
min_sync_level	int	0x0	force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)
mve_has_atomic_ticks	bool	0x0	Enable atomic ticks behaviour for vector instructions flagged as such (e.g. VLDR)
number_of_itm_stimulus_ports	int	0x20	The number of ITM stimulus ports
rd_ns_bus_err_behave	int	0x1	External read aborts in nonsecure domain 0:ignored, 1:precise, 2:imprecise, 3=imprecise except SO.
rd_s_bus_err_behave	int	0x1	External read aborts in secure domain 0:ignored, 1:precise, 2:imprecise, 3=imprecise except SO.
scheduler_mode	int	0x0	Control the interleaving of instructions in this processor (0=default long quantum, 1=low latency mode, short quantum and signal checking, 2=lock-breaking mode, long quantum with additional context switches near load-exclusive instructions, 3=ISSCompare)
semihosting-Thumb_SVC	int	0xab	T32 SVC number for semihosting
semihosting-cmd_line	string	""	Command line available to semihosting SVC calls
semihosting-cwd	string	""	Base directory for semihosting file access.
semihosting-enable	bool	0x1	Enable semihosting SVC traps. Applications that do not use semihosting must set this parameter to false.
semihosting-heap_base	int	0x0	Virtual address of heap base
semihosting-heap_limit	int	0x10700000	Virtual address of top of heap
semihosting-prefix	bool	0x0	Prefix semihosting output with target instance name

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-84 Parameters for ARM_AEMv8M (continued)

Name	Type	Default value	Description
semihosting-stack_base	int	0x10700000	Virtual address of base of descending stack
semihosting-stack_limit	int	0x10800000	Virtual address of stack limit
vector_fetch_busfault_sets_HFSR_FORCED	bool	0x0	Only v8.0M. Set HFSR.FORCED when a vector table read generates a HardFault
vfp-enable_at_reset	bool	0x0	Enable VFP in CPACR, CPPWR, NSACR at reset. Warning: Arm recommends going through the implementation's suggested VFP power-up sequence!
vfp-present	bool	0x1	Set whether the model has VFP support
wr_ns_bus_err_behave	int	0x3	External write aborts in nonsecure domain 0:ignored, 1:precise, 2:imprecise, 3=imprecise except SO.
wr_s_bus_err_behave	int	0x3	External write aborts in secure domain 0:ignored, 1:precise, 2:imprecise, 3=imprecise except SO.

3.4.5 ARMCortexA5CT

ARMCortexA5CT CPU component. This model is written in C++ and models version r0p0 of the RTL.

ARMCortexA5CT contains the following CADI targets:

- ARM_Cortex-A5UP
- Cluster_ARM_Cortex-A5UP
- PVCache
- TlbCadi

ARMCortexA5CT contains the following MTI components:

- *4.10 ARM_Cortex-A5 - trace* on page 4-1127
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402
- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.108 PVCache - trace* on page 4-2761
- *4.114 TLB - trace* on page 4-2834

About ARMCortexA5CT

- The `ase-present` and `vfp-present` parameters configure the synthesis options:

  vfp present and ase present
  Neon™ and VFPv3-D32 are supported.

  vfp present and ase not present
  VFPv3-D16 is supported.

  vfp not present and ase present
  Illegal. Forces `vfp-present` to true so model has Neon and VFPv3-D32 support.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

vfp not present and ase not present
Model has neither Neon nor VFPv3-D32 support.

- Specify `false` for the `device-accurate-tlb` parameter to enable modeling a different number of TLBs if this improves simulation performance. In this case, the simulation is architecturally-accurate, but not device-accurate. Architectural accuracy is almost always sufficient. Specify `true` if device accuracy is required.

Differences between the model and the RTL

This component has the following differences from the corresponding revision of the RTL implementation:

- The RR bit in the SCTLR is ignored.
- The Power Control Register in the system control coprocessor is implemented but writing to it does not change the behavior of the model.
- This component implements L1 cache as architecturally defined, but does not implement L2 cache. If you require an L2 cache you can add a PL310 Level 2 Cache Controller component.

Table 3-85 Ports

Name	Protocol	Type	Description
`cfgend[1]`	*Signal* on page 2-85	Slave	This signal if for EE bit initialisation.
`cfgnmfi[1]`	*Signal* on page 2-85	Slave	This signal disables FIQ mask in CPSR.
`clk_in`	`ClockSignal`	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the core executes instructions.
`clusterid`	*Value* on page 2-85	Slave	The port reads the value in CPU ID register field, bits[11:8] of the MPIDR.
`cp15sdisable[1]`	*Signal* on page 2-85	Slave	This signal disables write access to some system control processor registers.
`event`	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE.
`fiq[1]`	*Signal* on page 2-85	Slave	This signal drives the CPU's fast-interrupt handling.
`irq[1]`	*Signal* on page 2-85	Slave	This signal drives the CPU's interrupt handling.
`pmuirq[1]`	*Signal* on page 2-85	Master	Interrupt signal from performance monitoring unit.
`pvbus_m0`	`PVBus`	Master	The core will generate bus requests on this port.
`reset[1]`	*Signal* on page 2-85	Slave	Raising this signal will put the core into reset mode.
`standbywfe[1]`	*Signal* on page 2-85	Master	This signal indicates if a core is in WFE state.
`standbywfi[1]`	*Signal* on page 2-85	Master	This signal indicates if a core is in WFI state.
`teinit[1]`	*Signal* on page 2-85	Slave	This signal provides default exception handling state.
`ticks[1]`	*InstructionCount* on page 2-84	Master	This port should be connected to one of the two ticks ports on a 'visualisation' component, in order to display a running instruction count.
`vinithi[1]`	*Signal* on page 2-85	Slave	This signal controls of the location of the exception vectors at reset.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-86 Parameters for ARM_Cortex-A5UP

Name	Type	Default value	Description
cpu0.CFGEND	bool	0x0	Initialize to BE8 endianness
cpu0.CFGNMFI	bool	0x0	Enable nonmaskable FIQ interrupts on startup
cpu0.CP15SDISABLE	bool	0x0	Initialize to disable access to some CP15 registers
cpu0.POWERCTLI	int	0x0	Default power control state for processor
cpu0.TEINIT	bool	0x0	T32 exception enable. The default has exceptions including reset handled in A32 state
cpu0.VINITHI	bool	0x0	Initialize with high vectors enabled
cpu0.ase-present	bool	0x1	Set whether model has NEON support
cpu0.dcache-size	int	0x8000	Set D-cache size in bytes
cpu0.icache-size	int	0x8000	Set I-cache size in bytes
cpu0.min_sync_level	int	0x0	force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)
cpu0.semihosting-ARM_HLT	int	0xf000	ARM HLT number for semihosting
cpu0.semihosting-ARM_SVC	int	0x123456	ARM SVC number for semihosting
cpu0.semihosting-Thumb_HLT	int	0x3c	Thumb HLT number for semihosting
cpu0.semihosting-Thumb_SVC	int	0xab	Thumb SVC number for semihosting
cpu0.semihosting-cmd_line	string	""	Command line available to semihosting SVC calls
cpu0.semihosting-cwd	string	""	Base directory for semihosting file access.
cpu0.semihosting-enable	bool	0x1	Enable semihosting SVC traps. Applications that do not use semihosting must set this parameter to false
cpu0.semihosting-heap_base	int	0x0	Virtual address of heap base
cpu0.semihosting-heap_limit	int	0xf000000	Virtual address of top of heap
cpu0.semihosting-hlt-enable	bool	0x0	Enable semihosting HLT traps. Applications that use HLT semihosting must set this parameter to true and the semihosting-enable parameter to true
cpu0.semihosting-prefix	bool	0x0	Prefix semihosting output with target instance name
cpu0.semihosting-stack_base	int	0x10000000	Virtual address of base of descending stack
cpu0.semihosting-stack_limit	int	0xf000000	Virtual address of stack limit
cpu0.vfp-enable_at_reset	bool	0x0	Enable coprocessor access and VFP at reset
cpu0.vfp-present	bool	0x1	Set whether the model has VFP support

Table 3-87 Parameters for Cluster_ARM_Cortex-A5UP

Name	Type	Default value	Description
CLUSTER_ID	int	0x0	Processor cluster ID value
cpi_div	int	0x1	Divider for calculating CPI (Cycles Per Instruction)
cpi_mul	int	0x1	Multiplier for calculating CPI (Cycles Per Instruction)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-87 Parameters for Cluster_ARM_Cortex-A5UP (continued)

Name	Type	Default value	Description
dcache-hit_latency	int	0x0	D-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
dcache-maintenance_latency	int	0x0	D-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
dcache-miss_latency	int	0x0	D-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
dcache-read_access_latency	int	0x0	D-Cache timing annotation latency for read accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
dcache-read_latency	int	0x0	D-Cache timing annotation latency for read accesses given in ticks per byte accessed.dcache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
dcache-state_modelled	bool	0x0	Set whether D-cache has stateful implementation
dcache-write_access_latency	int	0x0	D-Cache timing annotation latency for write accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-write_latency is set. This is only used when dcache-state_modelled=true.
dcache-write_latency	int	0x0	D-Cache timing annotation latency for write accesses given in ticks per byte accessed. dcache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
device-accurate-tlb	bool	0x0	Specify whether all TLBs are modeled
icache-hit_latency	int	0x0	I-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when icache-state_modelled=true.
icache-maintenance_latency	int	0x0	I-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when icache-state_modelled=true.
icache-miss_latency	int	0x0	I-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when icache-state_modelled=true.
icache-read_access_latency	int	0x0	I-Cache timing annotation latency for read accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if icache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when icache-state_modelled=true.
icache-read_latency	int	0x0	I-Cache timing annotation latency for read accesses given in ticks per byte accessed.icache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when icache-state_modelled=true.
icache-state_modelled	bool	0x0	Set whether I-cache has stateful implementation

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

3.4.6 ARMCortexA5MPx1CT

ARMCortexA5MPx1CT CPU component. This model is written in C++ and models version r0p0 of the RTL.

ARMCortexA5MPx1CT contains the following CADI targets:

- ARM_Cortex-A5MP
- Cluster_ARM_Cortex-A5MP
- PVCache
- TlbCadi

ARMCortexA5MPx1CT contains the following MTI components:

- *4.14 ARM_Cortex-A5MP - trace* on page 4-1266
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402
- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.108 PVCache - trace* on page 4-2761
- *4.114 TLB - trace* on page 4-2834

About ARMCortexA5MPx1CT

- The following components also exist:
  - ARMCortexA5MPx2CT.
  - ARMCortexA5MPx4CT.

  The per-core parameters are preceded by `cpu`*n*`.`, where *n* identifies the core (0-3).
- The `ase-present` and `vfp-present` parameters configure the synthesis options:

  vfp present and ase present
  Neon and VFPv3-D32 are supported.

  vfp present and ase not present
  VFPv3-D16 is supported.

  vfp not present and ase present
  Illegal. Forces `vfp-present` to true so model has Neon and VFPv3-D32 support.

  vfp not present and ase not present
  Model has neither Neon nor VFPv3-D32 support.
- Specify `false` for the `device-accurate-tlb` parameter to enable modeling a different number of TLBs if this improves simulation performance. In this case, the simulation is architecturally-accurate, but not device-accurate. Architectural accuracy is almost always sufficient. Specify `true` if device accuracy is required.
- If you are using the ARMCortexA5MPx*n*CT component in a VE platform model, the `PERIPHBASE` parameter is set automatically to `0x2C000000` and is not visible in the parameter list.
- The `vfp-enable_at_reset` option is model-specific and has no hardware equivalent. Arm recommends that it is only used in test systems and tied off to `false` in production systems.

Differences between the model and the RTL

This component has the following differences from the corresponding revision of the RTL implementation:

- There is a single memory port combining instruction, data, DMA and peripheral access.
- The GIC does not respect the CFGSDISABLE signal. This leads to some registers being accessible when they must not be.
- The SCU enable bit is ignored. The SCU is always enabled.
- The SCU ignores the invalidate all register.
- The Broadcast TLB or cache operations in this model do not cause other cores in the cluster that are asleep because of *Wait For Interrupt* (WFI) to wake up.
- The RR bit in the SCTLR is ignored.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

- The Power Control Register in the system control coprocessor is implemented but writing to it does not change the behavior of the model.
- When modeling the SCU, coherency operations are represented by a memory write followed by a read to refill from memory, rather than using cache-to-cache transfers.
- This component implements L1 cache as architecturally defined, but does not implement L2 cache. If you require an L2 cache you can add a PL310 Level 2 Cache Controller component.

Table 3-88 Ports

Name	Protocol	Type	Description
acp_s	PVBus	Slave	AXI ACP slave port.
cfgend[1]	*Signal* on page 2-85	Slave	This signal if for EE bit initialisation.
cfgnmfi[1]	*Signal* on page 2-85	Slave	This signal disables FIQ mask in CPSR.
cfgsdisable	*Signal* on page 2-85	Slave	This signal disables write access to some secure Interrupt Controller registers.
clk_in	ClockSignal	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the core executes instructions.
clusterid	*Value* on page 2-85	Slave	The port reads the value in CPU ID register field, bits[11:8] of the MPIDR.
cp15sdisable[1]	*Signal* on page 2-85	Slave	This signal disables write access to some system control processor registers.
event	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE.
filteren	*Signal* on page 2-85	Slave	This signal enables filtering of address ranges between master bus ports.
filterend	*Value* on page 2-85	Slave	This port sets end of region mapped to pvbus_m1.
filterstart	*Value* on page 2-85	Slave	This port sets start of region mapped to pvbus_m1.
fiq[1]	*Signal* on page 2-85	Slave	This signal drives the CPU's fast-interrupt handling.
fiqout[1]	*Signal* on page 2-85	Master	This signal exports internal VGIC FIQ signal to the CPU.
ints[224]	*Signal* on page 2-85	Slave	This signal drives the CPU's fast-interrupt handling.
irq[1]	*Signal* on page 2-85	Slave	This signal drives the CPU's interrupt handling.
irqout[1]	*Signal* on page 2-85	Master	This signal exports internal VGIC IRQ signal to the CPU.
periphbase	*Value* on page 2-85	Slave	This port sets the base address of private peripheral region.
periphclk_in	ClockSignal	Slave	The timer and the watchdog take need a clk that is scaled down atleast by factor of two.
periphreset	*Signal* on page 2-85	Slave	This signal resets timer and interrupt controller.
pmuirq[1]	*Signal* on page 2-85	Master	Interrupt signal from performance monitoring unit.
pvbus_m0	PVBus	Master	AXI master 0 bus master channel.
pvbus_m1	PVBus	Master	AXI master 1 bus master channel.
pwrctli[1]	*Value* on page 2-85	Slave	This port sets reset value for scu CPU status register.
pwrctlo[1]	*Value* on page 2-85	Master	This port sends scu CPU status register bits.
reset[1]	*Signal* on page 2-85	Slave	Raising this signal will put the core into reset mode.
scureset	*Signal* on page 2-85	Slave	This signal resets SCU.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-88 Ports (continued)

Name	Protocol	Type	Description
smpnamp[1]	*Signal* on page 2-85	Master	This signals AMP or SMP mode for each Cortex-A5 processor.
standbywfe[1]	*Signal* on page 2-85	Master	This signal indicates if a core is in WFE state.
standbywfi[1]	*Signal* on page 2-85	Master	This signal indicates if a core is in WFI state.
teinit[1]	*Signal* on page 2-85	Slave	This signal provides default exception handling state.
ticks[1]	*InstructionCount* on page 2-84	Master	This port should be connected to one of the two ticks ports on a 'visualisation' component, in order to display a running instruction count.
vinithi[1]	*Signal* on page 2-85	Slave	This signal controls of the location of the exception vectors at reset.
virtio_s	PVBus	Slave	The virtio coherent port, hooks directly into the L2 system and becomes coherent (assuming attributes are set correctly).
wdreset[1]	*Signal* on page 2-85	Slave	This signal resets individual watchdog.
wdresetreq[1]	*Signal* on page 2-85	Master	This signal resets rest of the CA5MP system.

Table 3-89 Parameters for ARM_Cortex-A5MP

Name	Type	Default value	Description
cpu0.CFGEND	bool	0x0	Initialize to BE8 endianness
cpu0.CFGNMFI	bool	0x0	Enable nonmaskable FIQ interrupts on startup
cpu0.CP15SDISABLE	bool	0x0	Initialize to disable access to some CP15 registers
cpu0.POWERCTLI	int	0x0	Default power control state for processor
cpu0.SMPnAMP	bool	0x0	Set whether the processor is part of a coherent domain
cpu0.TEINIT	bool	0x0	T32 exception enable. The default has exceptions including reset handled in A32 state
cpu0.VINITHI	bool	0x0	Initialize with high vectors enabled
cpu0.ase-present	bool	0x1	Set whether model has NEON support
cpu0.dcache-size	int	0x8000	Set D-cache size in bytes
cpu0.icache-size	int	0x8000	Set I-cache size in bytes
cpu0.min_sync_level	int	0x0	force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)
cpu0.semihosting-ARM_HLT	int	0xf000	ARM HLT number for semihosting
cpu0.semihosting-ARM_SVC	int	0x123456	ARM SVC number for semihosting
cpu0.semihosting-Thumb_HLT	int	0x3c	Thumb HLT number for semihosting
cpu0.semihosting-Thumb_SVC	int	0xab	Thumb SVC number for semihosting
cpu0.semihosting-cmd_line	string	""	Command line available to semihosting SVC calls
cpu0.semihosting-cwd	string	""	Base directory for semihosting file access.
cpu0.semihosting-enable	bool	0x1	Enable semihosting SVC traps. Applications that do not use semihosting must set this parameter to false

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-89 Parameters for ARM_Cortex-A5MP (continued)

Name	Type	Default value	Description
cpu0.semihosting-heap_base	int	0x0	Virtual address of heap base
cpu0.semihosting-heap_limit	int	0xf000000	Virtual address of top of heap
cpu0.semihosting-hlt-enable	bool	0x0	Enable semihosting HLT traps. Applications that use HLT semihosting must set this parameter to true and the semihosting-enable parameter to true
cpu0.semihosting-prefix	bool	0x0	Prefix semihosting output with target instance name
cpu0.semihosting-stack_base	int	0x10000000	Virtual address of base of descending stack
cpu0.semihosting-stack_limit	int	0xf000000	Virtual address of stack limit
cpu0.vfp-enable_at_reset	bool	0x0	Enable coprocessor access and VFP at reset
cpu0.vfp-present	bool	0x1	Set whether the model has VFP support

Table 3-90 Parameters for Cluster_ARM_Cortex-A5MP

Name	Type	Default value	Description
CFGSDISABLE	bool	0x0	Disable some accesses to GIC registers
CLUSTER_ID	int	0x0	Processor cluster ID value
FILTEREN	bool	0x0	Enable filtering of accesses through pvbus_m0
FILTEREND	int	0x0	End of region filtered to pvbus_m0
FILTERSTART	int	0x0	Base of region filtered to pvbus_m0
PERIPHBASE	int	0x13080000	Base address of peripheral memory space
cpi_div	int	0x1	Divider for calculating CPI (Cycles Per Instruction)
cpi_mul	int	0x1	Multiplier for calculating CPI (Cycles Per Instruction)
dcache-hit_latency	int	0x0	D-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
dcache-maintenance_latency	int	0x0	D-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
dcache-miss_latency	int	0x0	D-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
dcache-read_access_latency	int	0x0	D-Cache timing annotation latency for read accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-90 Parameters for Cluster_ARM_Cortex-A5MP (continued)

Name	Type	Default value	Description
dcache-read_latency	int	0x0	D-Cache timing annotation latency for read accesses given in ticks per byte accessed.dcache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
dcache-snoop_data_transfer_latency	int	0x0	D-Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
dcache-state_modelled	bool	0x0	Set whether D-cache has stateful implementation
dcache-write_access_latency	int	0x0	D-Cache timing annotation latency for write accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-write_latency is set. This is only used when dcache-state_modelled=true.
dcache-write_latency	int	0x0	D-Cache timing annotation latency for write accesses given in ticks per byte accessed. dcache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
device-accurate-tlb	bool	0x0	Specify whether all TLBs are modeled
dic-spi_count	int	0x40	Number of shared peripheral interrupts implemented
icache-hit_latency	int	0x0	I-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when icache-state_modelled=true.
icache-maintenance_latency	int	0x0	I-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when icache-state_modelled=true.
icache-miss_latency	int	0x0	I-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when icache-state_modelled=true.
icache-read_access_latency	int	0x0	I-Cache timing annotation latency for read accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if icache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when icache-state_modelled=true.
icache-read_latency	int	0x0	I-Cache timing annotation latency for read accesses given in ticks per byte accessed.icache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when icache-state_modelled=true.
icache-state_modelled	bool	0x0	Set whether I-cache has stateful implementation

3.4.7 ARMCortexA7x1CT

ARMCortexA7x1CT CPU component. This model is written in C++ and models version r0p0 of the RTL.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ARMCortexA7x1CT contains the following CADI targets:

- ARM_Cortex-A7
- Cluster_ARM_Cortex-A7
- PVCache
- TlbCadi

ARMCortexA7x1CT contains the following MTI components:

- *4.17 ARM_Cortex-A7 - trace* on page 4-1367
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402
- *4.71 GICv2 - trace* on page 4-2470
- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.108 PVCache - trace* on page 4-2761
- *4.114 TLB - trace* on page 4-2834

About ARMCortexA7x1CT

- The following components also exist:
  - ARMCortexA7x2CT.
  - ARMCortexA7x3CT.
  - ARMCortexA7x4CT.

  The per-core parameters are preceded by `cpu`*`n`*`.`, where *n* identifies the core (0-3).
- The `ase-present` and `vfp-present` parameters configure the synthesis options:

  vfp present and ase present
  Neon and VFPv4-D32 are supported.

  vfp present and ase not present
  VFPv4-D16 is supported.

  vfp not present and ase present
  Illegal. Forces `vfp-present` to true so model has Neon and VFPv4-D32 support.

  vfp not present and ase not present
  Model has neither Neon nor VFPv4-D32 support.
- If you are using the ARMCortexA7x*n*CT component in a VE platform model, the `PERIPHBASE` parameter is set automatically to `0x2C000000` and is not visible in the parameter list.
- The `vfp-enable_at_reset` option is model-specific and has no hardware equivalent. Arm recommends that it is only used in test systems and tied off to `false` in production systems.

ACE limitation

AXI Coherency Extensions (ACE) are extensions to AXI4 that support system-level cache-coherency between multiple clusters. The ACE cache models in the Cortex-A15 and the Cortex-A7, and the ACE support in the CCI-400 have the limitation that they only process one transaction at a time. Normally, the simulation processes each transaction to completion before allowing any master to generate another transaction. However, in the following situation the simulation might fail. If a SystemC bus slave calls `wait()` while it is processing a transaction, this call might allow another master to issue another transaction that passes through the CCI-400 or the Cortex-A15/Cortex-A7 caches. This situation could happen if a SystemC bus master running in another thread is connected to one of the ACE-lite ports on the CCI-400.

Differences between the CT model and RTL implementations

This model has the following differences from the corresponding revision of the RTL implementation:

- The GIC does not respect the CFGSDISABLE signal. This leads to some registers wrongly being accessible.
- The Broadcast *Translation Lookaside Buffer* (TLB) or cache operations in this model do not cause other cores in the cluster that are asleep because of *Wait For Interrupt* (WFI) to wake up.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

- The model ignores the RR bit in the SCTLR.
- The model implements the Power Control Register in the system control coprocessor but writing to it does not change the behavior of the model.
- The model does not implement ETM registers.
- The model does not support the Cortex-A7 mechanism to read the internal memory that the Cache and TLB structures use through the implementation defined region of the system coprocessor interface.
- The model does not upgrade DCIMVAC operations to DCCIMVAC.

Table 3-91 Ports

Name	Protocol	Type	Description
CNTHPIRQ[4]	*Signal* on page 2-85	Master	Outputs of the generic timers.
CNTPNSIRQ[4]	*Signal* on page 2-85	Master	Outputs of the generic timers.
CNTPSIRQ[4]	*Signal* on page 2-85	Master	Outputs of the generic timers.
CNTVIRQ[4]	*Signal* on page 2-85	Master	Outputs of the generic timers.
axierrirq	*Signal* on page 2-85	Master	Imprecise aborts from the L2 are signaled by pulsing this pin, typically they are connect to an interrupt controller.
broadcastcachemaint	*Signal* on page 2-85	Slave	Enable broadcasting of cache maintenance operations to downstream caches.
broadcastinner	*Signal* on page 2-85	Slave	Enable broadcasting of Inner Shareable transactions.
broadcastouter	*Signal* on page 2-85	Slave	Enable broadcasting of Outer Shareable transactions.
cfgend[4]	*Signal* on page 2-85	Slave	This signal controls the SCTLR.EE bit.
cfgsdisable	*Signal* on page 2-85	Slave	This signal disables write access to some secure Interrupt Controller registers.
clk_in	ClockSignal	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the core executes instructions.
clusterid	*Value* on page 2-85	Slave	This port sets the value in the CLUSTERID field (bits[11:8]) of the MPIDR.
cntvalueb	*CounterInterface* on page 2-84	Slave	Interface to SoC level counter module.
cp15sdisable[4]	*Signal* on page 2-85	Slave	This signal disables write access to some secure system control processor registers.
cpuporeset[4]	*Signal* on page 2-85	Slave	Signal initializes all processor logic including NEON, VFP, Debug, PTM, breakpoint and watchpoint.
event	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE.
fiq[4]	*Signal* on page 2-85	Slave	This signal drives the CPU's fast-interrupt handling.
fiqout[4]	*Signal* on page 2-85	Master	This signal exports internal VGIC FIQ signal to the CPU..
irq[4]	*Signal* on page 2-85	Slave	This signal drives the CPU's interrupt handling.
irqout[4]	*Signal* on page 2-85	Master	This signal exports internal VGIC IRQ signal to the CPU.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-91 Ports (continued)

Name	Protocol	Type	Description
irqs[480]	*Signal* on page 2-85	Slave	These signals drive the CPU's interrupt controller interrupt lines.
l2reset	*Signal* on page 2-85	Slave	This signal resets the shared L2 memory system, interrupt controller and timer logic.
periphbase	*Value_64* on page 2-85	Slave	This port sets the base address of the private peripheral region.
pmuirq[4]	*Signal* on page 2-85	Master	Interrupt signal from performance monitoring unit.
presetdbg	*Signal* on page 2-85	Slave	Signal initializes the shared Debug APB, CTI and CTM logic.
pvbus_m0	PVBus	Master	The core will generate bus requests on this port.
reset[4]	*Signal* on page 2-85	Slave	Raising this signal will put the core into reset mode.
standbywfe[4]	*Signal* on page 2-85	Master	This signal indicates if a core is in WFE state.
standbywfi[4]	*Signal* on page 2-85	Master	This signal indicates if a core is in WFI state.
standbywfil2	*Signal* on page 2-85	Master	Indicate that all the individual processors and the L2 memory system are in a WFI state.
teinit[4]	*Signal* on page 2-85	Slave	This signal enables Thumb exceptions (controls the SCTLR.TE bit).
ticks[4]	*InstructionCount* on page 2-84	Master	This port should be connected to one of the two ticks ports on a 'visualisation' component, in order to display a running instruction count.
vfiq[4]	*Signal* on page 2-85	Slave	Virtual FIQ inputs. Note that the fiq pins are wired directly to the core if there is no internal VGIC. If there is an internal VGIC then these are ignored.
vinithi[4]	*Signal* on page 2-85	Slave	This signal controls the location of the exception vectors at reset.
virq[4]	*Signal* on page 2-85	Slave	Virtual IRQ inputs. Note that the irq pins are wired directly to the core if there is no internal VGIC. If there is an internal VGIC then these are ignored.
virtio_s	PVBus	Slave	The virtio coherent port, hooks directly into the L2 system and becomes coherent (assuming attributes are set correctly).

Table 3-92 Parameters for Cluster_ARM_Cortex-A7

Name	Type	Default value	Description
BROADCASTCACHEMAINT	bool	0x1	Enable broadcasting of cache maintenance operations to downstream caches. The broadcastcachemaint signal will override this value if used.
BROADCASTINNER	bool	0x1	Enable broadcasting of Inner Shareable transactions. The broadcastinner signal will override this value if used.
BROADCASTOUTER	bool	0x1	Enable broadcasting of Outer Shareable transactions. The broadcastouter signal will override this value if used.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-92 Parameters for Cluster_ARM_Cortex-A7 (continued)

Name	Type	Default value	Description
CFGSDISABLE	bool	0x0	Disable some accesses to GIC registers
CLUSTER_ID	int	0x0	Processor cluster ID value
PERIPHBASE	int	0x13080000	Base address of peripheral memory space
cpi_div	int	0x1	Divider for calculating CPI (Cycles Per Instruction)
cpi_mul	int	0x1	Multiplier for calculating CPI (Cycles Per Instruction)
dic-spi_count	int	0x40	Number of shared peripheral interrupts implemented
disable_periph_decoder	bool	0x0	Disable memory mapped access to gic system registers
internal_vgic	bool	0x1	Configures whether the model of the processor contains a VGIC
l1_dcache-hit_latency	int	0x0	L1 D-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
l1_dcache-maintenance_latency	int	0x0	L1 D-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
l1_dcache-miss_latency	int	0x0	L1 D-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
l1_dcache-read_access_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l1_dcache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
l1_dcache-read_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per byte accessed.l1_dcache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
l1_dcache-snoop_data_transfer_latency	int	0x0	L1 D-Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
l1_dcache-state_modelled	bool	0x0	Set whether L1 D-cache has stateful implementation
l1_dcache-write_access_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l1_dcache-write_latency is set. This is only used when dcache-state_modelled=true.
l1_dcache-write_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per byte accessed. l1_dcache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-92 Parameters for Cluster_ARM_Cortex-A7 (continued)

Name	Type	Default value	Description
l1_icache-hit_latency	int	0x0	L1 I-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
l1_icache-maintenance_latency	int	0x0	L1 I-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
l1_icache-miss_latency	int	0x0	L1 I-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
l1_icache-read_access_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l1_icache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
l1_icache-read_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per byte accessed.l1_icache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
l1_icache-state_modelled	bool	0x0	Set whether L1 I-cache has stateful implementation
l2_cache-hit_latency	int	0x0	L2 Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
l2_cache-maintenance_latency	int	0x0	L2 Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
l2_cache-miss_latency	int	0x0	L2 Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
l2_cache-read_access_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2_cache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
l2_cache-read_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per byte accessed.l2_cache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
l2_cache-size	int	0x80000	Set L2 cache size in bytes

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-92 Parameters for Cluster_ARM_Cortex-A7 (continued)

Name	Type	Default value	Description
l2_cache-snoop_data_transfer_latency	int	0x0	L2 Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
l2_cache-snoop_issue_latency	int	0x0	L2 Cache timing annotation latency for snoop accesses issued by this cache in total ticks. This is only used when dcache-state_modelled=true.
l2_cache-state_modelled	bool	0x0	Set whether L2 cache has stateful implementation
l2_cache-write_access_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2_cache-write_latency is set. This is only used when dcache-state_modelled=true.
l2_cache-write_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per byte accessed. l2_cache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.

Table 3-93 Parameters for ARM_Cortex-A7

Name	Type	Default value	Description
cpu0.CFGEND	bool	0x0	Initialize to BE8 endianness
cpu0.CP15SDISABLE	bool	0x0	Initialize to disable access to some CP15 registers
cpu0.DBGROMADDR	int	0x12000003	This value is used to initialize the CP15 DBGDRAR register. Bits[39:12] of this register specify the ROM table physical address
cpu0.DBGROMADDRV	bool	0x1	If true this sets bits[1:0] of the CP15 DBGDRAR to indicate that the address is valid
cpu0.DBGSELFADDR	int	0x10003	This value is used to initialize the CP15 DBGDSAR register. Bits[39:17] of this register specify the ROM table physical address
cpu0.DBGSELFADDRV	bool	0x1	If true this sets bits[1:0] of the CP15 DBGDSAR to indicate that the address is valid
cpu0.TEINIT	bool	0x0	T32 exception enable. The default has exceptions including reset handled in A32 state
cpu0.VINITHI	bool	0x0	Initialize with high vectors enabled
cpu0.ase-present	bool	0x1	Set whether CT model has been built with NEON support
cpu0.l1_dcache-size	int	0x8000	Size of L1 D-cache
cpu0.l1_icache-size	int	0x8000	Size of L1 I-cache
cpu0.min_sync_level	int	0x0	Force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)
cpu0.semihosting-ARM_HLT	int	0xf000	ARM HLT number for semihosting
cpu0.semihosting-ARM_SVC	int	0x123456	ARM SVC number for semihosting
cpu0.semihosting-Thumb_HLT	int	0x3c	Thumb HLT number for semihosting
cpu0.semihosting-Thumb_SVC	int	0xab	Thumb SVC number for semihosting

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-93 Parameters for ARM_Cortex-A7 (continued)

Name	Type	Default value	Description
cpu0.semihosting-cmd_line	string	""	Command line available to semihosting SVC calls
cpu0.semihosting-cwd	string	""	Base directory for semihosting file access.
cpu0.semihosting-enable	bool	0x1	Enable semihosting SVC traps. Applications that do not use semihosting must set this parameter to false
cpu0.semihosting-heap_base	int	0x0	Virtual address of heap base
cpu0.semihosting-heap_limit	int	0xf000000	Virtual address of top of heap
cpu0.semihosting-hlt-enable	bool	0x0	Enable semihosting HLT traps. Applications that use HLT semihosting must set this parameter to true and the semihosting-enable parameter to true
cpu0.semihosting-prefix	bool	0x0	Prefix semihosting output with target instance name
cpu0.semihosting-stack_base	int	0x10000000	Virtual address of base of descending stack
cpu0.semihosting-stack_limit	int	0xf000000	Virtual address of stack limit
cpu0.vfp-enable_at_reset	bool	0x0	Enable coprocessor access and VFP at reset
cpu0.vfp-present	bool	0x1	Set whether CT model has been built with VFP support

3.4.8 ARMCortexA8CT

ARMCortexA8CT CPU component. This model is written in C++ and models version r2p1 of the RTL.

ARMCortexA8CT contains the following CADI targets:

- ARM_Cortex-A8
- PVCache
- TlbCadi

ARMCortexA8CT contains the following MTI components:

- *4.26 ARM_Cortex-A8 - trace* on page 4-1712
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402
- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.108 PVCache - trace* on page 4-2761
- *4.114 TLB - trace* on page 4-2834

About ARMCortexA8CT

- Specify `false` for the `device-accurate-tlb` parameter to enable modeling a different number of TLBs if this improves simulation performance. In this case, the simulation is architecturally-accurate, but not device-accurate. Architectural accuracy is almost always sufficient. Specify `true` if device accuracy is required.
- If the `l1_dcache-state_modelled` parameter is true, then `l2_cache-state_modelled` must also be true.
- When you use the `semihosting-cmd_line` parameter to set the command line that is available to semihosting SVC calls, the value of `argv[0]` is set to the first command-line argument, not to the name of an image.
- This component provides the registers that the *Technical Reference Manual* (TRM) specifies except for the coprocessor 14 registers, the integration and test registers, and the PLE model, which is register based and has no implemented behavior.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

These TLB registers do not have working implementations:
— D-TLB ATTR read/write.
— D-TLB CAM read/write.
— D-TLB PA read/write.
— Normal memory remap register.
— Primary memory remap register.

Differences between the model and the RTL

This component has the following differences from the corresponding revision of the RTL implementation:

- There is a single memory port combining instruction, data, DMA and peripheral access.
- The L2 cache write allocate policy is not configurable. It defaults to write-allocate. Writes to the configuration register succeed but are ignored, meaning that data can be unexpectedly stored in the L2 cache.
- Unaligned accesses with the MMU disabled on the processor do not cause data aborts.
- The `vfp-enable_at_reset` option is model-specific and has no hardware equivalent. Arm recommends that it is only used in test systems and tied off to `false` in production systems.
- ECC and parity schemes are hardware-specific so are not supported.

Table 3-94 Ports

Name	Protocol	Type	Description
`cfgend0`	*Signal* on page 2-85	Slave	Configure BE8 mode after a reset.
`cfgnmfi`	*Signal* on page 2-85	Slave	Configure FIQs as non-maskable after a reset.
`cfgte`	*Signal* on page 2-85	Slave	Configure exceptions to be taken in thumb mode after a reset.
`clk_in`	`ClockSignal`	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the core executes instructions.
`dmaexterrirq`	*Signal* on page 2-85	Master	L1 PLE error interrupt.
`dmairq`	*Signal* on page 2-85	Master	Interrupt signal from L1 PLE.
`dmasirq`	*Signal* on page 2-85	Master	Secure interrupt signal from L1 PLE.
`fiq`	*Signal* on page 2-85	Slave	This signal drives the CPU's fast-interrupt handling.
`irq`	*Signal* on page 2-85	Slave	This signal drives the CPU's interrupt handling.
`pmuirq`	*Signal* on page 2-85	Master	Interrupt signal from performance monitoring unit.
`pvbus_m`	`PVBus`	Master	The core will generate bus requests on this port.
`reset`	*Signal* on page 2-85	Slave	Raising this signal will put the core into reset mode.
`ticks`	*InstructionCount* on page 2-84	Master	Port allowing the number of instructions since startup to be read from the CPU.
`vinithi`	*Signal* on page 2-85	Slave	Configure high vectors after a reset.

Table 3-95 Parameters for ARM_Cortex-A8

Name	Type	Default value	Description
`CFGEND0`	`bool`	`0x0`	Initialize to BE8 endianness
`CFGNMFI`	`bool`	`0x0`	Enable nonmaskable FIQ interrupts on startup
`CFGTE`	`bool`	`0x0`	Initialize to take exceptions in T32 state. Model starts in T32 state

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-95 Parameters for ARM_Cortex-A8 (continued)

Name	Type	Default value	Description
CP15SDISABLE	bool	0x0	Initialize to disable access to some CP15 registers
VINITHI	bool	0x0	Initialize with high vectors enabled
cpi_div	int	0x1	Divider for calculating CPI (Cycles Per Instruction)
cpi_mul	int	0x1	Multiplier for calculating CPI (Cycles Per Instruction)
device-accurate-tlb	bool	0x0	Specify whether all TLBs are modeled
implements_vfp	bool	0x1	Set whether the model has been built with VFP and NEON support
l1_dcache-hit_latency	int	0x0	L1 D-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
l1_dcache-maintenance_latency	int	0x0	L1 D-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
l1_dcache-miss_latency	int	0x0	L1 D-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
l1_dcache-read_access_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l1_dcache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
l1_dcache-read_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per byte accessed.l1_dcache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
l1_dcache-size	int	0x8000	Set L1 D-cache size in bytes
l1_dcache-state_modelled	bool	0x0	Include Level 1 data cache state model
l1_dcache-write_access_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l1_dcache-write_latency is set. This is only used when dcache-state_modelled=true.
l1_dcache-write_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per byte accessed. l1_dcache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
l1_icache-hit_latency	int	0x0	L1 I-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-95 Parameters for ARM_Cortex-A8 (continued)

Name	Type	Default value	Description
l1_icache-miss_latency	int	0x0	L1 I-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
l1_icache-read_access_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l1_icache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
l1_icache-read_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per byte accessed.l1_icache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
l1_icache-size	int	0x8000	Set L1 I-cache size in bytes
l1_icache-state_modelled	bool	0x0	Include Level 1 instruction cache state model
l2_cache-hit_latency	int	0x0	L2 Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
l2_cache-maintenance_latency	int	0x0	L2 Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
l2_cache-miss_latency	int	0x0	L2 Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
l2_cache-read_access_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2_cache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
l2_cache-read_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per byte accessed.l2_cache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
l2_cache-size	int	0x40000	Set L2 cache size in bytes
l2_cache-snoop_data_transfer_latency	int	0x0	L2 Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-95 Parameters for ARM_Cortex-A8 (continued)

Name	Type	Default value	Description
l2_cache-state_modelled	bool	0x0	Include unified Level 2 cache state model
l2_cache-write_access_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2_cache-write_latency is set. This is only used when dcache-state_modelled=true.
l2_cache-write_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per byte accessed. l2_cache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
master_id	int	0x0	Master ID presented in bus transactions
min_sync_level	int	0x0	Force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)
semihosting-ARM_HLT	int	0xf000	ARM HLT number for semihosting
semihosting-ARM_SVC	int	0x123456	ARM SVC number for semihosting
semihosting-Thumb_HLT	int	0x3c	Thumb HLT number for semihosting
semihosting-Thumb_SVC	int	0xab	Thumb SVC number for semihosting
semihosting-cmd_line	string	""	Command line available to semihosting SVC calls
semihosting-cwd	string	""	Base directory for semihosting file access.
semihosting-enable	bool	0x1	Enable semihosting SVC traps. Applications that do not use semihosting must set this parameter to false
semihosting-heap_base	int	0x0	Virtual address of heap base
semihosting-heap_limit	int	0xf000000	Virtual address of top of heap
semihosting-hlt-enable	bool	0x0	Enable semihosting HLT traps. Applications that use HLT semihosting must set this parameter to true and the semihosting-enable parameter to true
semihosting-prefix	bool	0x0	Prefix semihosting output with target instance name
semihosting-stack_base	int	0x10000000	Virtual address of base of descending stack
semihosting-stack_limit	int	0xf000000	Virtual address of stack limit
siliconID	int	0x41000000	Value as read by the system coprocessor siliconID register
vfp-enable_at_reset	bool	0x0	Enable coprocessor access and VFP at reset

3.4.9 ARMCortexA9MPx1CT

ARMCortexA9MPx1CT CPU component. This model is written in C++ and models version r3p0 of the RTL.

ARMCortexA9MPx1CT contains the following CADI targets:

- ARM_Cortex-A9MP
- Cluster_ARM_Cortex-A9MP
- PVCache
- TlbCadi

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ARMCortexA9MPx1CT contains the following MTI components:

- *4.27 ARM_Cortex-A9MP - trace* on page 4-1736
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402
- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.108 PVCache - trace* on page 4-2761
- *4.114 TLB - trace* on page 4-2834

About ARMCortexA9MPx1CT

- The following components also exist:
  - ARMCortexA9MPx2CT.
  - ARMCortexA9MPx4CT.

  The per-core parameters are preceded by `cpu`*n*`.`, where *n* identifies the core (0-3).
- The `ase-present` and `vfp-present` parameters configure the synthesis options:

  vfp present and ase present
  Neon and VFPv3-D32 are supported.

  vfp present and ase not present
  VFPv3-D16 is supported.

  vfp not present and ase present
  Illegal. Forces `vfp-present` to true so model has Neon and VFPv3-D32 support.

  vfp not present and ase not present
  Model has neither Neon nor VFPv3-D32 support.
- If you are using the ARMCortexA9MPx*n*CT component in a VE platform model, the `PERIPHBASE` parameter is set automatically to `0x2C000000` and is not visible in the parameter list.
- When you use the `semihosting-cmd_line` parameter to set the command line that is available to semihosting SVC calls, the value of `argv[0]` is set to the first command-line argument, not to the name of an image.
- Specify `false` for the `device-accurate-tlb` parameter to enable modeling a different number of TLBs if this improves simulation performance. In this case, the simulation is architecturally-accurate, but not device-accurate. Architectural accuracy is almost always sufficient. Specify `true` if device accuracy is required.
- This component implements L1 cache as architecturally defined, but does not implement L2 cache. If you require L2 cache, you can add a PL310 Level 2 Cache Controller component.

Differences between the model and the RTL

This component has the following differences from the corresponding revision of the RTL implementation:

- The GIC does not respect the CFGSDISABLE signal. This leads to some registers being accessible when they must not be.
- The SCU enable bit is ignored. The SCU is always enabled.
- The SCU ignores the invalidate all register.
- The Broadcast TLB or cache operations in this model do not cause other cores in the cluster that are asleep because of WFI to wake up.
- The RR bit in the SCTLR is ignored.
- The Power Control Register in the system control coprocessor is implemented but writing to it does not change the behavior of the model.
- The model cannot be configured with a 128-entry TLB.
- When modeling the SCU, coherency operations are represented by a memory write followed by a read to refill from memory, rather than using cache-to-cache transfers.
- The `vfp-enable_at_reset` option is model-specific and has no hardware equivalent. Arm recommends that it is only used in test systems and tied off to `false` in production systems.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

- The following TLB registers do not have working implementations:
  — Main TLB Attr.
  — Main TLB PA.
  — Main TLB VA.
  — Normal memory remap register.
  — Primary memory remap register.
  — Read Main TLB Entry.
  — Write Main TLB Entry.

  In addition, the simulation does not distinguish peripheral accesses from data accesses, so it ignores configuration of the peripheral port memory remap register.
- Parity error support is hardware-specific so is not modeled.

Table 3-96 Ports

Name	Protocol	Type	Description
acp_s	PVBus	Slave	AXI ACP slave port.
cfgend[4]	*Signal* on page 2-85	Slave	This signal if for EE bit initialisation.
cfgnmfi[4]	*Signal* on page 2-85	Slave	This signal disables FIQ mask in CPSR.
cfgsdisable	*Signal* on page 2-85	Slave	This signal disables write access to some secure Interrupt Controller registers.
clk_in	ClockSignal	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the core executes instructions.
clusterid	*Value* on page 2-85	Slave	The port reads the value in CPU ID register field, bits[11:8] of the MPIDR.
cp15sdisable[4]	*Signal* on page 2-85	Slave	This signal disables write access to some system control processor registers.
event	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE.
filteren	*Signal* on page 2-85	Slave	This signal enables filtering of address ranges between master bus ports.
filterend	*Value* on page 2-85	Slave	This port sets end of region mapped to pvbus_m1.
filterstart	*Value* on page 2-85	Slave	This port sets start of region mapped to pvbus_m1.
fiq[4]	*Signal* on page 2-85	Slave	This signal drives the CPU's fast-interrupt handling.
fiqout[4]	*Signal* on page 2-85	Master	This signal exports internal VGIC FIQ signal to the CPU..
ints[224]	*Signal* on page 2-85	Slave	This signal drives the CPU's fast-interrupt handling.
irq[4]	*Signal* on page 2-85	Slave	This signal drives the CPU's interrupt handling.
irqout[4]	*Signal* on page 2-85	Master	This signal exports internal VGIC IRQ signal to the CPU.
periphbase	*Value* on page 2-85	Slave	This port sets the base address of private peripheral region.
periphclk_in	ClockSignal	Slave	The timer and the watchdog take need a clk that is scaled down atleast by factor of two.
periphreset	*Signal* on page 2-85	Slave	This signal resets timer and interrupt controller.
pmuirq[4]	*Signal* on page 2-85	Master	Interrupt signal from performance monitoring unit.
pvbus_m0	PVBus	Master	AXI master 0 bus master channel.
pvbus_m1	PVBus	Master	AXI master 1 bus master channel.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-96 Ports (continued)

Name	Protocol	Type	Description
pwrctli[4]	*Value* on page 2-85	Slave	This port sets reset value for scu CPU status register.
pwrctlo[4]	*Value* on page 2-85	Master	This port sends scu CPU status register bits.
reset[4]	*Signal* on page 2-85	Slave	Raising this signal will put the core into reset mode.
scureset	*Signal* on page 2-85	Slave	This signal resets SCU.
smpnamp[4]	*Signal* on page 2-85	Master	This signals AMP or SMP mode for each Cortex-A9 processor.
standbywfe[4]	*Signal* on page 2-85	Master	This signal indicates if a core is in WFE state.
standbywfi[4]	*Signal* on page 2-85	Master	This signal indicates if a core is in WFI state.
teinit[4]	*Signal* on page 2-85	Slave	This signal provides default exception handling state.
ticks[4]	*InstructionCount* on page 2-84	Master	This port should be connected to one of the two ticks ports on a 'visualisation' component, in order to display a running instruction count.
vinithi[4]	*Signal* on page 2-85	Slave	This signal controls of the location of the exception vectors at reset.
virtio_s	PVBus	Slave	The virtio coherent port, hooks directly into the L2 system and becomes coherent (assuming attributes are set correctly).
wdreset[4]	*Signal* on page 2-85	Slave	This signal resets individual watchdog.
wdresetreq[4]	*Signal* on page 2-85	Master	This signal resets rest of the CA9MP system.

Table 3-97 Parameters for ARM_Cortex-A9MP

Name	Type	Default value	Description
cpu0.CFGEND	bool	0x0	Initialize to BE8 endianness
cpu0.CFGNMFI	bool	0x0	Enable nonmaskable FIQ interrupts on startup
cpu0.CP15SDISABLE	bool	0x0	Initialize to disable access to some CP15 registers
cpu0.POWERCTLI	int	0x0	Default power control state for processor
cpu0.SMPnAMP	bool	0x0	Set whether the processor is part of a coherent domain
cpu0.TEINIT	bool	0x0	T32 exception enable. The default has exceptions including reset handled in A32 state
cpu0.VINITHI	bool	0x0	Initialize with high vectors enabled
cpu0.ase-present	bool	0x1	Set whether model has NEON support
cpu0.dcache-size	int	0x8000	Set D-cache size in bytes
cpu0.icache-size	int	0x8000	Set I-cache size in bytes
cpu0.min_sync_level	int	0x0	Force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)
cpu0.semihosting-ARM_HLT	int	0xf000	ARM HLT number for semihosting
cpu0.semihosting-ARM_SVC	int	0x123456	ARM SVC number for semihosting
cpu0.semihosting-Thumb_HLT	int	0x3c	Thumb HLT number for semihosting

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-97 Parameters for ARM_Cortex-A9MP (continued)

Name	Type	Default value	Description
cpu0.semihosting-Thumb_SVC	int	0xab	Thumb SVC number for semihosting
cpu0.semihosting-cmd_line	string	""	Command line available to semihosting SVC calls
cpu0.semihosting-cwd	string	""	Base directory for semihosting file access.
cpu0.semihosting-enable	bool	0x1	Enable semihosting SVC traps. Applications that do not use semihosting must set this parameter to false
cpu0.semihosting-heap_base	int	0x0	Virtual address of heap base
cpu0.semihosting-heap_limit	int	0xf000000	Virtual address of top of heap
cpu0.semihosting-hlt-enable	bool	0x0	Enable semihosting HLT traps. Applications that use HLT semihosting must set this parameter to true and the semihosting-enable parameter to true
cpu0.semihosting-prefix	bool	0x0	Prefix semihosting output with target instance name
cpu0.semihosting-stack_base	int	0x10000000	Virtual address of base of descending stack
cpu0.semihosting-stack_limit	int	0xf000000	Virtual address of stack limit
cpu0.vfp-enable_at_reset	bool	0x0	Enable coprocessor access and VFP at reset
cpu0.vfp-present	bool	0x1	Set whether the model has VFP support

Table 3-98 Parameters for Cluster_ARM_Cortex-A9MP

Name	Type	Default value	Description
CFGSDISABLE	bool	0x0	Disable some accesses to GIC registers
CLUSTER_ID	int	0x0	Processor cluster ID value
FILTEREN	bool	0x0	Enable filtering of accesses through pvbus_m0
FILTEREND	int	0x0	End of region filtered to pvbus_m0
FILTERSTART	int	0x0	Base of region filtered to pvbus_m0
PERIPHBASE	int	0x13080000	Base address of peripheral memory space
cpi_div	int	0x1	Divider for calculating CPI (Cycles Per Instruction)
cpi_mul	int	0x1	Multiplier for calculating CPI (Cycles Per Instruction)
dcache-hit_latency	int	0x0	D-cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
dcache-maintenance_latency	int	0x0	D-cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
dcache-miss_latency	int	0x0	D-cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-98 Parameters for Cluster_ARM_Cortex-A9MP (continued)

Name	Type	Default value	Description
dcache-read_access_latency	int	0x0	D-cache timing annotation latency for read accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
dcache-read_latency	int	0x0	D-cache timing annotation latency for read accesses given in ticks per byte accessed.dcache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
dcache-snoop_data_transfer_latency	int	0x0	D-Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
dcache-state_modelled	bool	0x0	Set whether D-cache has stateful implementation
dcache-write_access_latency	int	0x0	D-cache timing annotation latency for write accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-write_latency is set. This is only used when dcache-state_modelled=true.
dcache-write_latency	int	0x0	D-cache timing annotation latency for write accesses given in ticks per byte accessed. dcache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
device-accurate-tlb	bool	0x0	Specify whether all TLBs are modeled
dic-spi_count	int	0x40	Number of shared peripheral interrupts implemented
icache-hit_latency	int	0x0	I-cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when icache-state_modelled=true.
icache-maintenance_latency	int	0x0	I-cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when icache-state_modelled=true.
icache-miss_latency	int	0x0	I-cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when icache-state_modelled=true.
icache-read_access_latency	int	0x0	I-cache timing annotation latency for read accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if icache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when icache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-98 Parameters for Cluster_ARM_Cortex-A9MP (continued)

Name	Type	Default value	Description
icache-read_latency	int	0x0	I-cache timing annotation latency for read accesses given in ticks per byte accessed.icache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when icache-state_modelled=true.
icache-state_modelled	bool	0x0	Set whether I-cache has stateful implementation

3.4.10 ARMCortexA9UPCT

ARMCortexA9UPCT CPU component. This model is written in C++ and models version r3p0 of the RTL.

ARMCortexA9UPCT contains the following CADI targets:

- ARM_Cortex-A9UP
- Cluster_ARM_Cortex-A9UP
- PVCache
- TlbCadi

ARMCortexA9UPCT contains the following MTI components:

- *4.28 ARM_Cortex-A9UP - trace* on page 4-1772
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402
- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.108 PVCache - trace* on page 4-2761
- *4.114 TLB - trace* on page 4-2834

About ARMCortexA9UPCT

- The `ase-present` and `vfp-present` parameters configure the synthesis options:

  vfp present and ase present
  Neon and VFPv3-D32 are supported.

  vfp present and ase not present
  VFPv3-D16 is supported.

  vfp not present and ase present
  Illegal. Forces `vfp-present` to `true` so model has Neon and VFPv3-D32 support.

  vfp not present and ase not present
  Model has neither Neon nor VFPv3-D32 support.
- Specify `false` for the `device-accurate-tlb` parameter to enable modeling a different number of TLBs if this improves simulation performance. In this case, the simulation is architecturally-accurate, but not device-accurate. Architectural accuracy is almost always sufficient. Specify `true` if device accuracy is required.
- When you use the `semihosting-cmd_line` parameter to set the command line that is available to semihosting SVC calls, the value of `argv[0]` is set to the first command-line argument, not to the name of an image.

Differences between the model and the RTL

This component has the following differences from the corresponding revision of the RTL implementation:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

- The RR bit in the SCTLR is ignored.
- The Power Control Register in the system control coprocessor is implemented but writing to it does not change the behavior of the model.
- The model cannot be configured with a 128-entry TLB.
- This component implements L1 cache as architecturally defined, but does not implement L2 cache. If you require L2 cache you can add a PL310 Level 2 Cache Controller component.
- Parity error support is hardware-specific so is not modeled.
- The `vfp-enable_at_reset` option is model-specific and has no hardware equivalent. Arm recommends that it is only used in test systems and tied off to `false` in production systems.

These TLB registers do not have working implementations:

- Normal memory remap register.
- Primary memory remap register.
- Read Main TLB Entry.
- Write Main TLB Entry.
- Main TLB VA.
- Main TLB PA.
- Main TLB Attr.

In addition, the simulation does not distinguish peripheral accesses from data accesses, so it ignores configuration of the peripheral port memory remap register.

Table 3-99 Ports

Name	Protocol	Type	Description
`cfgend[1]`	*Signal* on page 2-85	Slave	This signal if for EE bit initialisation.
`cfgnmfi[1]`	*Signal* on page 2-85	Slave	This signal disables FIQ mask in CPSR.
`clk_in`	`ClockSignal`	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the core executes instructions.
`clusterid`	*Value* on page 2-85	Slave	The port reads the value in CPU ID register field, bits[11:8] of the MPIDR.
`cp15sdisable[1]`	*Signal* on page 2-85	Slave	This signal disables write access to some system control processor registers.
`event`	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE.
`fiq[1]`	*Signal* on page 2-85	Slave	This signal drives the CPU's fast-interrupt handling.
`irq[1]`	*Signal* on page 2-85	Slave	This signal drives the CPU's interrupt handling.
`pmuirq[1]`	*Signal* on page 2-85	Master	Interrupt signal from performance monitoring unit.
`pvbus_m0`	`PVBus`	Master	The core will generate bus requests on this port.
`reset[1]`	*Signal* on page 2-85	Slave	Raising this signal will put the core into reset mode.
`standbywfe[1]`	*Signal* on page 2-85	Master	This signal indicates if a core is in WFE state.
`standbywfi[1]`	*Signal* on page 2-85	Master	This signal indicates if a core is in WFI state.
`teinit[1]`	*Signal* on page 2-85	Slave	This signal provides default exception handling state.
`ticks[1]`	*InstructionCount* on page 2-84	Master	This port should be connected to one of the two ticks ports on a 'visualisation' component, in order to display a running instruction count.
`vinithi[1]`	*Signal* on page 2-85	Slave	This signal controls of the location of the exception vectors at reset.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-100 Parameters for ARM_Cortex-A9UP

Name	Type	Default value	Description
cpu0.CFGEND	bool	0x0	Initialize to BE8 endianness
cpu0.CFGNMFI	bool	0x0	Enable nonmaskable FIQ interrupts on startup
cpu0.CP15SDISABLE	bool	0x0	Initialize to disable access to some CP15 registers
cpu0.POWERCTLI	int	0x0	Default power control state for processor
cpu0.TEINIT	bool	0x0	T32 exception enable. The default has exceptions including reset handled in A32 state
cpu0.VINITHI	bool	0x0	Initialize with high vectors enabled
cpu0.ase-present	bool	0x1	Set whether model has NEON support
cpu0.dcache-size	int	0x8000	Set D-cache size in bytes
cpu0.icache-size	int	0x8000	Set I-cache size in bytes
cpu0.min_sync_level	int	0x0	Force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)
cpu0.semihosting-ARM_HLT	int	0xf000	ARM HLT number for semihosting
cpu0.semihosting-ARM_SVC	int	0x123456	ARM SVC number for semihosting
cpu0.semihosting-Thumb_HLT	int	0x3c	Thumb HLT number for semihosting
cpu0.semihosting-Thumb_SVC	int	0xab	Thumb SVC number for semihosting
cpu0.semihosting-cmd_line	string	""	Command line available to semihosting SVC calls
cpu0.semihosting-cwd	string	""	Base directory for semihosting file access.
cpu0.semihosting-enable	bool	0x1	Enable semihosting SVC traps. Applications that do not use semihosting must set this parameter to false
cpu0.semihosting-heap_base	int	0x0	Virtual address of heap base
cpu0.semihosting-heap_limit	int	0xf000000	Virtual address of top of heap
cpu0.semihosting-hlt-enable	bool	0x0	Enable semihosting HLT traps. Applications that use HLT semihosting must set this parameter to true and the semihosting-enable parameter to true
cpu0.semihosting-prefix	bool	0x0	Prefix semihosting output with target instance name
cpu0.semihosting-stack_base	int	0x10000000	Virtual address of base of descending stack
cpu0.semihosting-stack_limit	int	0xf000000	Virtual address of stack limit
cpu0.vfp-enable_at_reset	bool	0x0	Enable coprocessor access and VFP at reset
cpu0.vfp-present	bool	0x1	Set whether the model has VFP support

Table 3-101 Parameters for Cluster_ARM_Cortex-A9UP

Name	Type	Default value	Description
CLUSTER_ID	int	0x0	Processor cluster ID value
cpi_div	int	0x1	Divider for calculating CPI (Cycles Per Instruction)
cpi_mul	int	0x1	Multiplier for calculating CPI (Cycles Per Instruction)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-101 Parameters for Cluster_ARM_Cortex-A9UP (continued)

Name	Type	Default value	Description
dcache-hit_latency	int	0x0	D-cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
dcache-maintenance_latency	int	0x0	D-cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
dcache-miss_latency	int	0x0	D-cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
dcache-read_access_latency	int	0x0	D-cache timing annotation latency for read accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
dcache-read_latency	int	0x0	D-cache timing annotation latency for read accesses given in ticks per byte accessed.dcache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
dcache-state_modelled	bool	0x0	Set whether D-cache has stateful implementation
dcache-write_access_latency	int	0x0	D-cache timing annotation latency for write accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-write_latency is set. This is only used when dcache-state_modelled=true.
dcache-write_latency	int	0x0	D-cache timing annotation latency for write accesses given in ticks per byte accessed. dcache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
device-accurate-tlb	bool	0x0	Specify whether all TLBs are modeled
icache-hit_latency	int	0x0	I-cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when icache-state_modelled=true.
icache-maintenance_latency	int	0x0	I-cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when icache-state_modelled=true.
icache-miss_latency	int	0x0	I-cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when icache-state_modelled=true.
icache-read_access_latency	int	0x0	I-cache timing annotation latency for read accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if icache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when icache-state_modelled=true.
icache-read_latency	int	0x0	I-cache timing annotation latency for read accesses given in ticks per byte accessed.icache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when icache-state_modelled=true.
icache-state_modelled	bool	0x0	Set whether I-cache has stateful implementation

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

3.4.11 ARMCortexA15x1CT

ARMCortexA15x1CT CPU component. This model is written in C++ and models version r2p0 of the RTL.

ARMCortexA15x1CT contains the following CADI targets:

- ARM_Cortex-A15
- Cluster_ARM_Cortex-A15
- PVCache
- TlbCadi

ARMCortexA15x1CT contains the following MTI components:

- *4.6 ARM_Cortex-A15 - trace* on page 4-979
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402
- *4.71 GICv2 - trace* on page 4-2470
- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.108 PVCache - trace* on page 4-2761
- *4.114 TLB - trace* on page 4-2834

About ARMCortexA15x1CT

- The following components also exist:
  - ARMCortexA15x2CT.
  - ARMCortexA15x3CT.
  - ARMCortexA15x4CT.

  The per-core parameters are preceded by `cpu`*n*`.`, where *n* identifies the core (0-3).
- The `ase-present` and `vfp-present` parameters configure the synthesis options:

  vfp present and ase present
  Neon and VFPv4-D32 are supported.

  vfp present and ase not present
  VFPv4-D16 is supported.

  vfp not present and ase present
  Illegal. Forces `vfp-present` to `true` so model has Neon and VFPv4-D32 support.

  vfp not present and ase not present
  Model has neither Neon nor VFPv4-D32 support.
- If you are using the ARMCortexA15x*n*CT component on a VE model platform, the `PERIPHBASE` parameter is set automatically to `0x2C000000` and is not visible in the parameter list.

ACE limitation

AXI Coherency Extensions (ACE) are extensions to AXI4 that support system-level cache-coherency between multiple clusters. The ACE cache models in the Cortex-A15 and the Cortex-A7, and the ACE support in the CCI-400 have the limitation that they only process one transaction at a time. Normally, the simulation processes each transaction to completion before allowing any master to generate another transaction. However, in the following situation the simulation might fail. If a SystemC bus slave calls `wait()` while it is processing a transaction, this call might allow another master to issue another transaction that passes through the CCI-400 or the Cortex-A15/Cortex-A7 caches. This situation could happen if a SystemC bus master running in another thread is connected to one of the ACE-lite ports on the CCI-400.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Differences between the model and the RTL

This component has the following differences from the corresponding revision of the RTL implementation:

- The GIC does not respect the CFGSDISABLE signal. This leads to some registers wrongly being accessible.
- The Broadcast *Translation Lookaside Buffer* (TLB) or cache operations in the model do not cause other cores in the cluster that are asleep because of *Wait For Interrupt* (WFI) to wake up.
- It ignores the RR bit in the SCTLR.
- It implements the Power Control Register in the system control coprocessor but writing to it does not change the behavior of the model.
- When modeling the SCU, coherency operations are by memory writes then reads to refill from memory, rather than cache-to-cache transfers.
- It does not implement ETM registers.
- It implements TLB bitmap registers as RAZ/WI.
- It does not support the Cortex-A15 mechanism to read the internal memory that the Cache and TLB structures use through the implementation defined region of the system coprocessor interface. This includes the RAM Index Register, IL1DATA Registers, DL1DATA Registers, and associated functionality.
- The `vfp-enable_at_reset` option is model-specific and has no hardware equivalent. Arm recommends that it is only used in test systems and tied off to `false` in production systems.
- ECC and parity schemes are hardware-specific so are not supported.

Table 3-102 Ports

Name	Protocol	Type	Description
CNTHPIRQ[4]	*Signal* on page 2-85	Master	Outputs of the generic timers.
CNTPNSIRQ[4]	*Signal* on page 2-85	Master	Outputs of the generic timers.
CNTPSIRQ[4]	*Signal* on page 2-85	Master	Outputs of the generic timers.
CNTVIRQ[4]	*Signal* on page 2-85	Master	Outputs of the generic timers.
acp_s	PVBus	Slave	AXI ACP slave port.
broadcastcachemaint	*Signal* on page 2-85	Slave	Enable broadcasting of cache maintenance operations to downstream caches.
broadcastinner	*Signal* on page 2-85	Slave	Enable broadcasting of Inner Shareable transactions.
broadcastouter	*Signal* on page 2-85	Slave	Enable broadcasting of Outer Shareable transactions.
cfgend[4]	*Signal* on page 2-85	Slave	This signal controls the SCTLR.EE bit.
cfgsdisable	*Signal* on page 2-85	Slave	This signal disables write access to some secure Interrupt Controller registers.
clk_in	ClockSignal	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the core executes instructions.
clusterid	*Value* on page 2-85	Slave	This port sets the value in the CLUSTERID field (bits[11:8]) of the MPIDR.
cntvalueb	*CounterInterface* on page 2-84	Slave	Interface to SoC level counter module.
cp15sdisable[4]	*Signal* on page 2-85	Slave	This signal disables write access to some secure system control processor registers.
cpuporeset[4]	*Signal* on page 2-85	Slave	Signal initializes all processor logic including NEON, VFP, Debug, PTM, breakpoint and watchpoint.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-102 Ports (continued)

Name	Protocol	Type	Description
event	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE.
fiq[4]	*Signal* on page 2-85	Slave	This signal drives the CPU's fast-interrupt handling.
fiqout[4]	*Signal* on page 2-85	Master	This signal exports internal VGIC FIQ signal to the CPU..
irq[4]	*Signal* on page 2-85	Slave	This signal drives the CPU's interrupt handling.
irqout[4]	*Signal* on page 2-85	Master	This signal exports internal VGIC IRQ signal to the CPU.
irqs[224]	*Signal* on page 2-85	Slave	These signals drive the CPU's interrupt controller interrupt lines.
l2reset	*Signal* on page 2-85	Slave	This signal resets the shared L2 memory system, interrupt controller and timer logic.
periphbase	*Value_64* on page 2-85	Slave	This port sets the base address of the private peripheral region.
pmuirq[4]	*Signal* on page 2-85	Master	Interrupt signal from performance monitoring unit.
presetdbg	*Signal* on page 2-85	Slave	Signal initializes the shared Debug APB, CTI and CTM logic.
pvbus_m0	PVBus	Master	The core will generate bus requests on this port.
reset[4]	*Signal* on page 2-85	Slave	Raising this signal will put the core into reset mode.
standbywfe[4]	*Signal* on page 2-85	Master	This signal indicates if a core is in WFE state.
standbywfi[4]	*Signal* on page 2-85	Master	This signal indicates if a core is in WFI state.
standbywfil2	*Signal* on page 2-85	Master	Indicate that all the individual processors and the L2 memory system are in a WFI state.
teinit[4]	*Signal* on page 2-85	Slave	This signal enables Thumb exceptions (controls the SCTLR.TE bit).
ticks[4]	*InstructionCount* on page 2-84	Master	This port should be connected to one of the two ticks ports on a 'visualisation' component, in order to display a running instruction count.
vfiq[4]	*Signal* on page 2-85	Slave	Virtual FIQ inputs. Note that the fiq pins are wired directly to the core if there is no internal VGIC. If there is an internal VGIC then these are ignored.
vinithi[4]	*Signal* on page 2-85	Slave	This signal controls the location of the exception vectors at reset.
virq[4]	*Signal* on page 2-85	Slave	Virtual IRQ inputs. Note that the irq pins are wired directly to the core if there is no internal VGIC. If there is an internal VGIC then these are ignored.
virtio_s	PVBus	Slave	The virtio coherent port, hooks directly into the L2 system and becomes coherent (assuming attributes are set correctly).

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-103 Parameters for Cluster_ARM_Cortex-A15

Name	Type	Default value	Description
BROADCASTCACHEMAINT	bool	0x1	Enable broadcasting of cache maintenance operations to downstream caches. The broadcastcachemaint signal will override this value if used.
BROADCASTINNER	bool	0x1	Enable broadcasting of Inner Shareable transactions. The broadcastinner signal will override this value if used.
BROADCASTOUTER	bool	0x1	Enable broadcasting of Outer Shareable transactions. The broadcastouter signal will override this value if used.
CFGSDISABLE	bool	0x0	Disable some accesses to GIC registers
CLUSTER_ID	int	0x0	Processor cluster ID value
IMINLN	bool	0x1	Instruction cache minimum line size: false=32 bytes, true=64 bytes
PERIPHBASE	int	0x13080000	Base address of peripheral memory space
cpi_div	int	0x1	Divider for calculating CPI (Cycles Per Instruction)
cpi_mul	int	0x1	Multiplier for calculating CPI (Cycles Per Instruction)
dic-spi_count	int	0x40	Number of shared peripheral interrupts implemented
disable_periph_decoder	bool	0x0	Disable memory mapped access to gic system registers
internal_vgic	bool	0x1	Configures whether the model of the processor contains a Virtualized Generic Interrupt Controller (VGIC)
l1_dcache-hit_latency	int	0x0	L1 D-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
l1_dcache-maintenance_latency	int	0x0	L1 D-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
l1_dcache-miss_latency	int	0x0	L1 D-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
l1_dcache-read_access_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l1_dcache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
l1_dcache-read_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per byte accessed.l1_dcache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
l1_dcache-snoop_data_transfer_latency	int	0x0	L1 D-Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
l1_dcache-state_modelled	bool	0x0	Set whether L1 D-cache has stateful implementation

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-103 Parameters for Cluster_ARM_Cortex-A15 (continued)

Name	Type	Default value	Description
l1_dcache-write_access_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l1_dcache-write_latency is set. This is only used when dcache-state_modelled=true.
l1_dcache-write_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per byte accessed. l1_dcache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
l1_icache-hit_latency	int	0x0	L1 I-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
l1_icache-maintenance_latency	int	0x0	L1 I-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
l1_icache-miss_latency	int	0x0	L1 I-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
l1_icache-read_access_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l1_icache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
l1_icache-read_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per byte accessed.l1_icache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
l1_icache-state_modelled	bool	0x0	Set whether L1 I-cache has stateful implementation
l2-data-slice	int	0x0	L2 data RAM slice
l2-tag-slice	int	0x0	L2 tag RAM slice
l2_cache-hit_latency	int	0x0	L2 Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
l2_cache-maintenance_latency	int	0x0	L2 Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
l2_cache-miss_latency	int	0x0	L2 Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-103 Parameters for Cluster_ARM_Cortex-A15 (continued)

Name	Type	Default value	Description
l2_cache-read_access_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2_cache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
l2_cache-read_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per byte accessed.l2_cache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
l2_cache-size	int	0x80000	Set L2 cache size in bytes
l2_cache-snoop_data_transfer_latency	int	0x0	L2 Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
l2_cache-snoop_issue_latency	int	0x0	L2 Cache timing annotation latency for snoop accesses issued by this cache in total ticks. This is only used when dcache-state_modelled=true.
l2_cache-state_modelled	bool	0x0	Set whether L2 cache has stateful implementation
l2_cache-write_access_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2_cache-write_latency is set. This is only used when dcache-state_modelled=true.
l2_cache-write_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per byte accessed. l2_cache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.

Table 3-104 Parameters for ARM_Cortex-A15

Name	Type	Default value	Description
cpu0.CFGEND	bool	0x0	Initialize to BE8 endianness
cpu0.CP15SDISABLE	bool	0x0	Initialize to disable access to some CP15 registers
cpu0.DBGROMADDR	int	0x12000003	This value is used to initialize the CP15 DBGDRAR register. Bits[39:12] of this register specify the ROM table physical address
cpu0.DBGROMADDRV	bool	0x1	If true, this sets bits[1:0] of the CP15 DBGDRAR to indicate that the address is valid
cpu0.DBGSELFADDR	int	0x10003	This value is used to initialize the CP15 DBGDSAR register. Bits[39:17] of this register specify the ROM table physical address
cpu0.DBGSELFADDRV	bool	0x1	If true, this sets bits[1:0] of the CP15 DBGDSAR to indicate that the address is valid
cpu0.TEINIT	bool	0x0	T32 exception enable. The default has exceptions including reset handled in A32 state

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-104 Parameters for ARM_Cortex-A15 (continued)

Name	Type	Default value	Description
cpu0.VINITHI	bool	0x0	Initialize with high vectors enabled
cpu0.ase-present	bool	0x1	Set whether CT model has been built with NEON support
cpu0.min_sync_level	int	0x0	Force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)
cpu0.semihosting-ARM_HLT	int	0xf000	ARM HLT number for semihosting
cpu0.semihosting-ARM_SVC	int	0x123456	ARM SVC number for semihosting
cpu0.semihosting-Thumb_HLT	int	0x3c	Thumb HLT number for semihosting
cpu0.semihosting-Thumb_SVC	int	0xab	Thumb SVC number for semihosting
cpu0.semihosting-cmd_line	string	""	Command line available to semihosting SVC calls
cpu0.semihosting-cwd	string	""	Base directory for semihosting file access.
cpu0.semihosting-enable	bool	0x1	Enable semihosting SVC traps. Applications that do not use semihosting must set this parameter to false
cpu0.semihosting-heap_base	int	0x0	Virtual address of heap base
cpu0.semihosting-heap_limit	int	0xf000000	Virtual address of top of heap
cpu0.semihosting-hlt-enable	bool	0x0	Enable semihosting HLT traps. Applications that use HLT semihosting must set this parameter to true and the semihosting-enable parameter to true
cpu0.semihosting-prefix	bool	0x0	Prefix semihosting output with target instance name
cpu0.semihosting-stack_base	int	0x10000000	Virtual address of base of descending stack
cpu0.semihosting-stack_limit	int	0xf000000	Virtual address of stack limit
cpu0.vfp-enable_at_reset	bool	0x0	Enable coprocessor access and VFP at reset
cpu0.vfp-present	bool	0x1	Set whether CT model has been built with VFP support

3.4.12 ARMCortexA17x1CT

ARMCortexA17x1CT CPU component. This model is written in C++ and models version r0p0 of the RTL.

ARMCortexA17x1CT contains the following CADI targets:

- ARM_Cortex-A17
- Cluster_ARM_Cortex-A17
- PVCache
- TlbCadi

ARMCortexA17x1CT contains the following MTI components:

- *4.7 ARM_Cortex-A17 - trace* on page 4-1015
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402
- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.108 PVCache - trace* on page 4-2761
- *4.114 TLB - trace* on page 4-2834

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

About ARMCortexA17x1CT

- The following components also exist:
  — ARMCortexA17x2CT.
  — ARMCortexA17x3CT.
  — ARMCortexA17x4CT.

  The per-core parameters are preceded by `cpu`*n*`.`, where *n* identifies the core (0-3).
- The `ase-present` and `vfp-present` parameters configure the synthesis options:

  vfp present and ase present
  NEON and VFPv4-D32 are supported.

  vfp present and ase not present
  VFPv4-D16 is supported.

  vfp not present and ase present
  Illegal. Forces `vfp-present` to `true` so model has NEON and VFPv4-D32 support.

  vfp not present and ase not present
  Model has neither NEON nor VFPv4-D32 support.
- This model exposes the `BROADCASTCACHEMAINT`, `BROADCASTINNER`, and `BROADCASTOUTER` parameters at the CPU level, rather than at the cluster level. To achieve correct behavior, set the same value for all CPUs in the cluster.
- The `vfp-enable_at_reset` option is model-specific and has no hardware equivalent. Arm recommends that it is only used in test systems and tied off to `false` in production systems.
- ECC support is hardware-specific so is not modeled.

Table 3-105 Ports

Name	Protocol	Type	Description
CFGADDRFILTENDNS	*Value_64* on page 2-85	Slave	NS end address (only bits 39:20 are used).
CFGADDRFILTENDS	*Value_64* on page 2-85	Slave	S end address (only bits 39:20 are used).
CFGADDRFILTENNS	*Signal* on page 2-85	Slave	Enable periph port filtering for NS accesses.
CFGADDRFILTENS	*Signal* on page 2-85	Slave	Enable periph port filtering for S accesses.
CFGADDRFILTSTARTNS	*Value_64* on page 2-85	Slave	NS start address (only bits 39:20 are used).
CFGADDRFILTSTARTS	*Value_64* on page 2-85	Slave	S start address (only bits 39:20 are used).
CNTHPIRQ[4]	*Signal* on page 2-85	Master	Outputs of the generic timers.
CNTPNSIRQ[4]	*Signal* on page 2-85	Master	Outputs of the generic timers.
CNTPSIRQ[4]	*Signal* on page 2-85	Master	Outputs of the generic timers.
CNTVIRQ[4]	*Signal* on page 2-85	Master	Outputs of the generic timers.
acp_s	PVBus	Slave	AXI ACP slave port.
broadcastcachemaint	*Signal* on page 2-85	Slave	Enable broadcasting of cache maintenance operations to downstream caches.
broadcastinner	*Signal* on page 2-85	Slave	Enable broadcasting of Inner Shareable transactions.
broadcastouter	*Signal* on page 2-85	Slave	Enable broadcasting of Outer Shareable transactions.
cfgend[4]	*Signal* on page 2-85	Slave	This signal controls the SCTLR.EE bit.
clk_in	ClockSignal	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the core executes instructions.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-105 Ports (continued)

Name	Protocol	Type	Description
clusterid	*Value* on page 2-85	Slave	This port sets the value in the CLUSTERID field (bits[11:8]) of the MPIDR.
cntvalueb	*CounterInterface* on page 2-84	Slave	Interface to SoC level counter module.
cp15sdisable[4]	*Signal* on page 2-85	Slave	This signal disables write access to some secure system control processor registers.
cpuporeset[4]	*Signal* on page 2-85	Slave	Signal initializes all processor logic including NEON, VFP, Debug, PTM, breakpoint and watchpoint.
event	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE.
fiq[4]	*Signal* on page 2-85	Slave	This signal drives the CPU's fast-interrupt handling.
irq[4]	*Signal* on page 2-85	Slave	This signal drives the CPU's interrupt handling.
l2flushdone	*Signal* on page 2-85	Master	Flush of L2 memory system complete.
l2flushreq	*Signal* on page 2-85	Slave	Request flush of L2 memory system.
l2reset	*Signal* on page 2-85	Slave	This signal resets the shared L2 memory system and timer logic.
peripheral_m	PVBus	Master	The core's peripheral port. Controlled by filter registers.
pmuirq[4]	*Signal* on page 2-85	Master	Interrupt signal from performance monitoring unit.
presetdbg	*Signal* on page 2-85	Slave	Signal initializes the shared Debug APB, CTI and CTM logic.
pvbus_m0	PVBus	Master	The core will generate bus requests on this port.
reset[4]	*Signal* on page 2-85	Slave	Raising this signal will put the core into reset mode.
standbywfe[4]	*Signal* on page 2-85	Master	This signal indicates if a core is in WFE state.
standbywfi[4]	*Signal* on page 2-85	Master	This signal indicates if a core is in WFI state.
standbywfil2	*Signal* on page 2-85	Master	Indicate that all the individual processors and the L2 memory system are in a WFI state.
teinit[4]	*Signal* on page 2-85	Slave	This signal enables Thumb exceptions (controls the SCTLR.TE bit).
ticks[4]	*InstructionCount* on page 2-84	Master	This port should be connected to one of the two ticks ports on a 'visualisation' component, in order to display a running instruction count.
vfiq[4]	*Signal* on page 2-85	Slave	Virtual FIQ inputs.
vinithi[4]	*Signal* on page 2-85	Slave	This signal controls the location of the exception vectors at reset.
virq[4]	*Signal* on page 2-85	Slave	Virtual IRQ inputs.
virtio_s	PVBus	Slave	The virtio coherent port, hooks directly into the L2 system and becomes coherent (assuming attributes are set correctly).

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-106 Parameters for Cluster_ARM_Cortex-A17

Name	Type	Default value	Description
CFGADDRFILTENDNS	int	0x0	Peripheral port NS end address
CFGADDRFILTENDS	int	0x0	Peripheral port S end address
CFGADDRFILTENNS	bool	0x0	Peripheral port NS address filtering enabled
CFGADDRFILTENS	bool	0x0	Peripheral port S address filtering enabled
CFGADDRFILTSTARTNS	int	0x0	Peripheral port NS start address
CFGADDRFILTSTARTS	int	0x0	Peripheral port S start address
CLUSTER_ID	int	0x0	Processor cluster ID value
IMINLN	bool	0x1	Instruction cache minimum line size: false=32 bytes, true=64 bytes
cpi_div	int	0x1	Divider for calculating CPI (Cycles Per Instruction)
cpi_mul	int	0x1	Multiplier for calculating CPI (Cycles Per Instruction)
l1_dcache-hit_latency	int	0x0	L1 D-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
l1_dcache-maintenance_latency	int	0x0	L1 D-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
l1_dcache-miss_latency	int	0x0	L1 D-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
l1_dcache-read_access_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l1_dcache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
l1_dcache-read_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per byte accessed.l1_dcache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
l1_dcache-snoop_data_transfer_latency	int	0x0	L1 D-Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
l1_dcache-state_modelled	bool	0x0	Set whether L1 D-cache has stateful implementation
l1_dcache-write_access_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l1_dcache-write_latency is set. This is only used when dcache-state_modelled=true.
l1_dcache-write_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per byte accessed. l1_dcache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-106 Parameters for Cluster_ARM_Cortex-A17 (continued)

Name	Type	Default value	Description
l1_icache-hit_latency	int	0x0	L1 I-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
l1_icache-maintenance_latency	int	0x0	L1 I-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
l1_icache-miss_latency	int	0x0	L1 I-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
l1_icache-read_access_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l1_icache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
l1_icache-read_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per byte accessed.l1_icache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
l1_icache-state_modelled	bool	0x0	Set whether L1 I-cache has stateful implementation
l2-data-slice	int	0x0	L2 data RAM slice
l2-tag-slice	int	0x0	L2 tag RAM slice
l2_cache-hit_latency	int	0x0	L2 Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
l2_cache-maintenance_latency	int	0x0	L2 Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
l2_cache-miss_latency	int	0x0	L2 Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
l2_cache-read_access_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2_cache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
l2_cache-read_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per byte accessed.l2_cache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
l2_cache-size	int	0x40000	Set L2 cache size in bytes

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-106 Parameters for Cluster_ARM_Cortex-A17 (continued)

Name	Type	Default value	Description
l2_cache-snoop_data_transfer_latency	int	0x0	L2 Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
l2_cache-snoop_issue_latency	int	0x0	L2 Cache timing annotation latency for snoop accesses issued by this cache in total ticks. This is only used when dcache-state_modelled=true.
l2_cache-state_modelled	bool	0x0	Set whether L2 cache has stateful implementation
l2_cache-write_access_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2_cache-write_latency is set. This is only used when dcache-state_modelled=true.
l2_cache-write_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per byte accessed. l2_cache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.

Table 3-107 Parameters for ARM_Cortex-A17

Name	Type	Default value	Description
cpu0.BROADCASTCACHEMAINT	bool	0x1	Enable broadcasting of cache maintenance operations to downstream caches. The broadcastcachemaint signal will override this value if used.
cpu0.BROADCASTINNER	bool	0x1	Enable broadcasting of Inner Shareable transactions. The broadcastinner signal will override this value if used.
cpu0.BROADCASTOUTER	bool	0x1	Enable broadcasting of Outer Shareable transactions. The broadcastouter signal will override this value if used.
cpu0.CFGEND	bool	0x0	Initialize to BE8 endianness
cpu0.CP15SDISABLE	bool	0x0	Initialize to disable access to some CP15 registers
cpu0.DBGROMADDR	int	0x12000003	This value is used to initialize the CP15 DBGDRAR register. Bits[39:12] of this register specify the ROM table physical address
cpu0.DBGROMADDRV	bool	0x1	If true, this sets bits[1:0] of the CP15 DBGDRAR to indicate that the address is valid
cpu0.DBGSELFADDR	int	0x10003	This value is used to initialize the CP15 DBGDSAR register. Bits[39:17] of this register specify the ROM table physical address
cpu0.DBGSELFADDRV	bool	0x1	If true, this sets bits[1:0] of the CP15 DBGDSAR to indicate that the address is valid
cpu0.TEINIT	bool	0x0	T32 exception enable. The default has exceptions including reset handled in A32 state
cpu0.VINITHI	bool	0x0	Initialize with high vectors enabled
cpu0.ase-present	bool	0x1	Set whether CT model has been built with NEON support
cpu0.l1_icache-size	int	0x8000	Size of L1 I-cache

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-107 Parameters for ARM_Cortex-A17 (continued)

Name	Type	Default value	Description
cpu0.min_sync_level	int	0x0	Force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)
cpu0.semihosting-ARM_HLT	int	0xf000	ARM HLT number for semihosting
cpu0.semihosting-ARM_SVC	int	0x123456	ARM SVC number for semihosting
cpu0.semihosting-Thumb_HLT	int	0x3c	Thumb HLT number for semihosting
cpu0.semihosting-Thumb_SVC	int	0xab	Thumb SVC number for semihosting
cpu0.semihosting-cmd_line	string	""	Command line available to semihosting SVC calls
cpu0.semihosting-cwd	string	""	Base directory for semihosting file access.
cpu0.semihosting-enable	bool	0x1	Enable semihosting SVC traps. Applications that do not use semihosting must set this parameter to false
cpu0.semihosting-heap_base	int	0x0	Virtual address of heap base
cpu0.semihosting-heap_limit	int	0xf000000	Virtual address of top of heap
cpu0.semihosting-hlt-enable	bool	0x0	Enable semihosting HLT traps. Applications that use HLT semihosting must set this parameter to true and the semihosting-enable parameter to true
cpu0.semihosting-prefix	bool	0x0	Prefix semihosting output with target instance name
cpu0.semihosting-stack_base	int	0x10000000	Virtual address of base of descending stack
cpu0.semihosting-stack_limit	int	0xf000000	Virtual address of stack limit
cpu0.vfp-enable_at_reset	bool	0x0	Enable coprocessor access and VFP at reset
cpu0.vfp-present	bool	0x1	Set whether CT model has been built with VFP support

3.4.13 ARMCortexA32x1CT

ARMCortexA32x1CT CPU component. This model is written in C++ and models version r0p0 of the RTL.

ARMCortexA32x1CT contains the following CADI targets:

- ARM_Cortex-A32
- Cluster_ARM_Cortex-A32
- PVCache
- TlbCadi

ARMCortexA32x1CT contains the following MTI components:

- *4.8 ARM_Cortex-A32 - trace* on page 4-1051
- *4.48 ARMv8Cluster - trace* on page 4-2401
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402
- *4.68 DSU - trace* on page 4-2463
- *4.73 GICv3CPUInterfaceDecoder - trace* on page 4-2489
- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.108 PVCache - trace* on page 4-2761
- *4.114 TLB - trace* on page 4-2834

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

About ARMCortexA32x1CT

- The following components also exist:
  — ARMCortexA32x2CT.
  — ARMCortexA32x3CT.
  — ARMCortexA32x4CT.

  The per-core parameters are preceded by `cpun.`, where *n* identifies the core (0-3).
- If either L1 cache is stateful, then the L2 cache is stateful. This is controlled by the `dcache-state_modelled` and `icache-state_modelled` parameters.
- The cache latency parameters are only effective when you enable cache-state modeling.
- Timing annotation for transactions downstream of the cache and TLB models propagates through the models.
- Although Neon support is optional for the Cortex-A32 processor, this model does not implement the `ase-present` parameter. Therefore, it is not possible to configure the model to not support Neon.
- This model supports the Armv8-A Cryptography Extensions, which requires the Crypto plug-in to be loaded. See *Crypto plug-in* on page 5-2909 for details.
- The `semihosting-cwd` parameter sets the CWD that is used for semihosting. The host operating system limits the maximum path length. The `semihosting-cwd` parameter does not provide any security. Software running on the model can access files outside this directory using relative paths containing `..` or using absolute paths.
- The system designer decides whether a debug APB ties the external debug view with other system views. In the model, use a `PVBusDecoder` to direct traffic to the correct port, `dev_debug_s` or `memorymapped_debug_s`.
- The `vfp-enable_at_reset` option is model-specific and has no hardware equivalent. Arm recommends that it is only used in test systems and tied off to `false` in production systems.
- ECC and parity schemes are hardware-specific so are not supported.

Table 3-108 Ports

Name	Protocol	Type	Description
`CNTHPIRQ[4]`	*Signal* on page 2-85	Master	Timer signals to SOC.
`CNTPNSIRQ[4]`	*Signal* on page 2-85	Master	Timer signals to SOC.
`CNTPSIRQ[4]`	*Signal* on page 2-85	Master	Timer signals to SOC.
`CNTVIRQ[4]`	*Signal* on page 2-85	Master	Timer signals to SOC.
`acp_s`	`PVBus`	Slave	AXI ACP slave port.
`broadcastcachemaint`	*Signal* on page 2-85	Slave	ACE defined pins.
`broadcastinner`	*Signal* on page 2-85	Slave	ACE defined pins.
`broadcastouter`	*Signal* on page 2-85	Slave	ACE defined pins.
`cfgend[4]`	*Signal* on page 2-85	Slave	This signal if for EE bit initialisation.
`cfgsdisable`	*Signal* on page 2-85	Slave	This signal disables write access to some secure Interrupt Controller registers.
`cfgte[4]`	*Signal* on page 2-85	Slave	This signal provides default exception handling state.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-108 Ports (continued)

Name	Protocol	Type	Description
clk_in	ClockSignal	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the core executes instructions.
clrexmonack	*Signal* on page 2-85	Master	Acknowledge handshake signal for the clrexmonreq signal
clrexmonreq	*Signal* on page 2-85	Slave	Signals the clearing of an external global exclusive monitor
clusterid	*Value* on page 2-85	Slave	The port reads the value in CPU ID register field, bits[11:8] of the MPIDR.
cntvalueb	*CounterInterface* on page 2-84	Slave	Interface to SoC level counter module.
commirq[4]	*Signal* on page 2-85	Master	Interrupt signal from debug communications channel.
commrx[4]	*Signal* on page 2-85	Master	Receive portion of Data Transfer Register full.
commtx[4]	*Signal* on page 2-85	Master	Transmit portion of Data Transfer Register empty.
cp15sdisable[4]	*Signal* on page 2-85	Slave	This signal disables write access to some system control processor registers.
cp15sdisable2[4]	*Signal* on page 2-85	Slave	-
cpuporeset[4]	*Signal* on page 2-85	Slave	CPU power on reset.
cryptodisable[4]	*Signal* on page 2-85	Slave	Disable cryptography extensions after reset.
cti[4]	*v8EmbeddedCrossTrigger_controlprotocol* on page 2-84	Master	Cross trigger matrix port
ctidbgirq[4]	*Signal* on page 2-85	Master	Cross trigger matrix port.
dbgack[4]	*Signal* on page 2-85	Master	External debug interface.
dbgen[4]	*Signal* on page 2-85	Slave	External debug interface.
dbgnopwrdwn[4]	*Signal* on page 2-85	Master	No power-down request.
dbgpwrupreq[4]	*Signal* on page 2-85	Master	Processor powerup request.
dev_debug_s	PVBus	Slave	External debug interface.
edbgrq[4]	*Signal* on page 2-85	Slave	External debug interface.
event	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-108 Ports (continued)

Name	Protocol	Type	Description
fiq[4]	*Signal* on page 2-85	Slave	This signal drives the CPUs fast-interrupt handling.
gicv3_redistributor_s[4]	*GICv3Comms* on page 2-84	Slave	GICv3 AXI-stream port.
irq[4]	*Signal* on page 2-85	Slave	This signal drives the CPUs interrupt handling.
l2flushdone	*Signal* on page 2-85	Master	Flush of L2 memory system complete.
l2flushreq	*Signal* on page 2-85	Slave	Request flush of L2 memory system.
l2reset	*Signal* on page 2-85	Slave	Level2 reset.
memorymapped_debug_s	PVBus	Slave	External debug interface.
niden[4]	*Signal* on page 2-85	Slave	External debug interface.
periphbase	*Value_64* on page 2-85	Slave	This port sets the base address of private peripheral region.
pmuirq[4]	*Signal* on page 2-85	Master	Interrupt signal from performance monitoring unit.
presetdbg	*Signal* on page 2-85	Slave	Debug reset.
pvbus_m0	PVBus	Master	The core will generate bus requests on this port.
rei[4]	*Signal* on page 2-85	Slave	Per core RAM Error Interrupt
reset[4]	*Signal* on page 2-85	Slave	Reset.
romaddr	*Value_64* on page 2-85	Slave	Debug ROM base address.
romaddrv	*Signal* on page 2-85	Slave	Debug ROM base address valid.
sei[4]	*Signal* on page 2-85	Slave	Per core System Error physical pins.
smpen[4]	*Signal* on page 2-85	Master	This signals AMP or SMP mode for each core.
spiden[4]	*Signal* on page 2-85	Slave	External debug interface.
spniden[4]	*Signal* on page 2-85	Slave	External debug interface.
standbywfe[4]	*Signal* on page 2-85	Master	This signal indicates if a core is in WFE state.
standbywfi[4]	*Signal* on page 2-85	Master	This signal indicates if a core is in WFI state
standbywfil2	*Signal* on page 2-85	Master	This signal indicated all cores and L2 are idles and in low power state.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-108 Ports (continued)

Name	Protocol	Type	Description
ticks[4]	*InstructionCount* on page 2-84	Master	This port should be connected to one of the two ticks ports on a 'visualisation' component, in order to display a running instruction count.
vcpumntirq[4]	*Signal* on page 2-85	Master	Interrupt signal for virtual CPU maintenance IRQ.
vfiq[4]	*Signal* on page 2-85	Slave	Virtualised FIQ.
vinithi[4]	*Signal* on page 2-85	Slave	This signal controls of the location of the exception vectors at reset.
virq[4]	*Signal* on page 2-85	Slave	Virtualised IRQ.
virtio_s	PVBus	Slave	The virtio coherent port, hooks directly into the L2 system and becomes coherent (assuming attributes are set correctly).
vsei[4]	*Signal* on page 2-85	Slave	Per core virtual System Error physical pins.

Table 3-109 Parameters for Cluster_ARM_Cortex-A32

Name	Type	Default value	Description
BROADCASTCACHEMAINT	bool	0x1	Enable broadcasting of cache maintenance operations to downstream caches. The broadcastcachemaint signal will override this value if used.
BROADCASTINNER	bool	0x1	Enable broadcasting of Inner Shareable transactions. The broadcastinner signal will override this value if used.
BROADCASTOUTER	bool	0x1	Enable broadcasting of Outer Shareable transactions. The broadcastouter signal will override this value if used.
CLUSTER_ID	int	0x0	Processor cluster ID value
DBGROMADDR	int	0x0	Initialization value of DBGDRAR register. Bits[47:12] of this register specify the ROM table physical address.
DBGROMADDRV	bool	0x0	If true, set bits[1:0] of the CP15 DBGDRAR to indicate that the address is valid
GICDISABLE	bool	0x1	Disable the new style GICv3 CPU interface in each core model. Should be left enabled unless the platform contains a GICv3 distributor.
PERIPHBASE	int	0x13080000	Base address of peripheral memory space

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-109 Parameters for Cluster_ARM_Cortex-A32 (continued)

Name	Type	Default value	Description
bus_type	int	0x0	Cosmetic change that changes reset value of L2ACTLR register. 0, ACE. 1, CHI. 2, AXI
cpi_div	int	0x1	Divider for calculating CPI (Cycles Per Instruction)
cpi_mul	int	0x1	Multiplier for calculating CPI (Cycles Per Instruction)
dcache-hit_latency	int	0x0	L1 D-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
dcache-maintenance_latency	int	0x0	L1 D-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
dcache-miss_latency	int	0x0	L1 D-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
dcache-prefetch_enabled	bool	0x0	Enable simulation of data cache prefetching. This is only used when dcache-state_modelled=true
dcache-read_access_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per access (of size dcache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
dcache-read_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per byte accessed.dcache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
dcache-size	int	0x8000	L1 D-Cache size in bytes.
dcache-snoop_data_transfer_latency	int	0x0	L1 D-Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
dcache-state_modelled	bool	0x0	Set whether D-cache has stateful implementation

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-109 Parameters for Cluster_ARM_Cortex-A32 (continued)

Name	Type	Default value	Description
dcache-write_access_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per access (of size dcache-write_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-write_latency is set. This is only used when dcache-state_modelled=true.
dcache-write_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per byte accessed. dcache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
enable_simulation_performance_optimizations	bool	0x1	With this option enabled, the model will run more quickly, but be less accurate to exact CPU behavior. The model will still be functionally accurate for software, but may increase differences seen between hardware behavior and model behavior for certain workloads (it changes the micro-architectural value of stage12_tlb_size parameter to 1024).
icache-hit_latency	int	0x0	L1 I-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when icache-state_modelled=true.
icache-maintenance_latency	int	0x0	L1 I-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when icache-state_modelled=true.
icache-miss_latency	int	0x0	L1 I-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when icache-state_modelled=true.
icache-prefetch_enabled	bool	0x0	Enable simulation of instruction cache prefetching. This is only used when icache-state_modelled=true.
icache-read_access_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per access (of size icache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if icache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when icache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-109 Parameters for Cluster_ARM_Cortex-A32 (continued)

Name	Type	Default value	Description
icache-read_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per byte accessed.icache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when icache-state_modelled=true.
icache-size	int	0x8000	L1 I-Cache size in bytes.
icache-state_modelled	bool	0x0	Set whether I-cache has stateful implementation
l2cache-hit_latency	int	0x0	L2 Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
l2cache-maintenance_latency	int	0x0	L2 Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
l2cache-miss_latency	int	0x0	L2 Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
l2cache-read_access_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
l2cache-read_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per byte accessed.l2cache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
l2cache-size	int	0x80000	L2 Cache size in bytes.
l2cache-snoop_data_transfer_latency	int	0x0	L2 Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
l2cache-snoop_issue_latency	int	0x0	L2 Cache timing annotation latency for snoop accesses issued by this cache in total ticks. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-109 Parameters for Cluster_ARM_Cortex-A32 (continued)

Name	Type	Default value	Description
l2cache-write_access_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-write_latency is set. This is only used when dcache-state_modelled=true.
l2cache-write_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per byte accessed. l2cache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
ptw_latency	int	0x0	Page table walker latency for TA (Timing Annotation), expressed in simulation ticks
tlb_latency	int	0x0	TLB latency for TA (Timing Annotation), expressed in simulation ticks
walk_cache_latency	int	0x0	Walk cache latency for TA (Timing Annotation), expressed in simulation ticks

Table 3-110 Parameters for ARM_Cortex-A32

Name	Type	Default value	Description
cpu0.CFGEND	bool	0x0	Endianness configuration at reset. 0, little endian. 1, big endian.
cpu0.CFGTE	bool	0x0	Instruction set state when resetting into AArch32. 0, A32. 1, T32.
cpu0.CP15SDISABLE	bool	0x0	Initialize to disable access to some CP15 registers
cpu0.CP15SDISABLE2	bool	0x0	Initialize to disable access to some CP15 registers
cpu0.CRYPTODISABLE	bool	0x0	Disable cryptographic features.
cpu0.VINITHI	bool	0x0	Reset value of SCTLR.V.
cpu0.enable_trace_special_hlt_imm16	bool	0x0	Enable usage of parameter trace_special_hlt_imm16
cpu0.max_code_cache_mb	int	0x100	Maximum size of the simulation code cache (MiB). For platforms with more than 2 cores this limit will be scaled down. (e.g 1/8 for 16 or more cores)
cpu0.min_sync_level	int	0x0	Force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)
cpu0.semihosting-A32_HLT	int	0xf000	A32 HLT number for semihosting calls.
cpu0.semihosting-ARM_SVC	int	0x123456	A32 SVC number for semihosting calls.
cpu0.semihosting-T32_HLT	int	0x3c	T32 HLT number for semihosting calls.
cpu0.semihosting-Thumb_SVC	int	0xab	T32 SVC number for semihosting calls.
cpu0.semihosting-cmd_line	string	""	Command line available to semihosting calls.
cpu0.semihosting-cwd	string	""	Base directory for semihosting file access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-110 Parameters for ARM_Cortex-A32 (continued)

Name	Type	Default value	Description
cpu0.semihosting-enable	bool	0x1	Enable semihosting SVC/HLT traps.
cpu0.semihosting-heap_base	int	0x0	Virtual address of heap base.
cpu0.semihosting-heap_limit	int	0xf000000	Virtual address of top of heap.
cpu0.semihosting-stack_base	int	0x10000000	Virtual address of base of descending stack.
cpu0.semihosting-stack_limit	int	0xf000000	Virtual address of stack limit.
cpu0.trace_special_hlt_imm16	int	0xf000	For this HLT number, IF enable_trace_special_hlt_imm16=true, skip performing usual HLT execution but call MTI trace if registered
cpu0.vfp-enable_at_reset	bool	0x0	Enable VFP in CPACR, CPPWR, NSACR at reset. Warning: Arm recommends going through the implementation's suggested VFP power-up sequence!
cpu0.vfp-present	bool	0x1	Set whether the model has VFP support

3.4.14 ARMCortexA35x1CT

ARMCortexA35x1CT CPU component. This model is written in C++ and models version r0p0 of the RTL.

ARMCortexA35x1CT contains the following CADI targets:

- ARM_Cortex-A35
- Cluster_ARM_Cortex-A35
- PVCache
- TlbCadi

ARMCortexA35x1CT contains the following MTI components:

- *4.9 ARM_Cortex-A35 - trace* on page 4-1089
- *4.48 ARMv8Cluster - trace* on page 4-2401
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402
- *4.68 DSU - trace* on page 4-2463
- *4.73 GICv3CPUInterfaceDecoder - trace* on page 4-2489
- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.108 PVCache - trace* on page 4-2761
- *4.114 TLB - trace* on page 4-2834

About ARMCortexA35x1CT

- The following components also exist:
  - ARMCortexA35x2CT.
  - ARMCortexA35x3CT.
  - ARMCortexA35x4CT.

  The per-core parameters are preceded by `cpu`*n*`.`, where *n* identifies the core (0-3).
- If either L1 cache is stateful, then the L2 cache is stateful. This is controlled by the `dcache-state_modelled` and `icache-state_modelled` parameters.
- The cache latency parameters are only effective when you enable cache-state modeling.
- Timing annotation for transactions downstream of the cache and TLB models propagates through the models.
- Although Neon support is optional for the Cortex-A35 processor, this model does not implement the `ase-present` parameter. Therefore, it is not possible to configure the model to not support Neon.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

- This model supports the Armv8-A Cryptography Extensions, which requires the Crypto plug-in to be loaded. See *Crypto plug-in* on page 5-2909 for details.
- The `semihosting-cwd` parameter sets the CWD that is used for semihosting. The host operating system limits the maximum path length. The `semihosting-cwd` parameter does not provide any security. Software running on the model can access files outside this directory using relative paths containing `..` or using absolute paths.
- The system designer decides whether a debug APB ties the external debug view with other system views. In the model, use a `PVBusDecoder` to direct traffic to the correct port, `dev_debug_s` or `memorymapped_debug_s`.
- The `vfp-enable_at_reset` option is model-specific and has no hardware equivalent. Arm recommends that it is only used in test systems and tied off to `false` in production systems.
- ECC and parity schemes are hardware-specific so are not supported.

Table 3-111 Ports

Name	Protocol	Type	Description
`CNTHPIRQ[4]`	*Signal* on page 2-85	Master	Timer signals to SOC.
`CNTPNSIRQ[4]`	*Signal* on page 2-85	Master	Timer signals to SOC.
`CNTPSIRQ[4]`	*Signal* on page 2-85	Master	Timer signals to SOC.
`CNTVIRQ[4]`	*Signal* on page 2-85	Master	Timer signals to SOC.
`aa64naa32[4]`	*Signal* on page 2-85	Slave	Register width after reset.
`acp_s`	`PVBus`	Slave	AXI ACP slave port.
`broadcastcachemaint`	*Signal* on page 2-85	Slave	ACE defined pins.
`broadcastinner`	*Signal* on page 2-85	Slave	ACE defined pins.
`broadcastouter`	*Signal* on page 2-85	Slave	ACE defined pins.
`cfgend[4]`	*Signal* on page 2-85	Slave	This signal if for EE bit initialisation.
`cfgsdisable`	*Signal* on page 2-85	Slave	This signal disables write access to some secure Interrupt Controller registers.
`cfgte[4]`	*Signal* on page 2-85	Slave	This signal provides default exception handling state.
`clk_in`	`ClockSignal`	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the core executes instructions.
`clrexmonack`	*Signal* on page 2-85	Master	Acknowledge handshake signal for the clrexmonreq signal
`clrexmonreq`	*Signal* on page 2-85	Slave	Signals the clearing of an external global exclusive monitor
`clusterid`	*Value* on page 2-85	Slave	The port reads the value in CPU ID register field, bits[11:8] of the MPIDR.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-111 Ports (continued)

Name	Protocol	Type	Description
cntvalueb	*CounterInterface* on page 2-84	Slave	Interface to SoC level counter module.
commirq[4]	*Signal* on page 2-85	Master	Interrupt signal from debug communications channel.
commrx[4]	*Signal* on page 2-85	Master	Receive portion of Data Transfer Register full.
commtx[4]	*Signal* on page 2-85	Master	Transmit portion of Data Transfer Register empty.
cp15sdisable[4]	*Signal* on page 2-85	Slave	This signal disables write access to some system control processor registers.
cp15sdisable2[4]	*Signal* on page 2-85	Slave	This signal disables write access to some system control processor registers.
cpuporeset[4]	*Signal* on page 2-85	Slave	CPU power on reset.
cryptodisable[4]	*Signal* on page 2-85	Slave	Disable cryptography extensions after reset.
cti[4]	*v8EmbeddedCrossTrigger_controlprotocol* on page 2-84	Master	Cross trigger matrix port.
ctidbgirq[4]	*Signal* on page 2-85	Master	Cross trigger matrix port.
dbgack[4]	*Signal* on page 2-85	Master	External debug interface.
dbgen[4]	*Signal* on page 2-85	Slave	External debug interface.
dbgnopwrdwn[4]	*Signal* on page 2-85	Master	No power-down request.
dbgpwrupreq[4]	*Signal* on page 2-85	Master	Processor powerup request.
dev_debug_s	PVBus	Slave	External debug interface.
edbgrq[4]	*Signal* on page 2-85	Slave	External debug interface.
event	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE.
fiq[4]	*Signal* on page 2-85	Slave	This signal drives the CPUs fast-interrupt handling.
gicv3_redistributor_s[4]	*GICv3Comms* on page 2-84	Slave	GICv3 AXI-stream port.
irq[4]	*Signal* on page 2-85	Slave	This signal drives the CPUs interrupt handling.
l2flushdone	*Signal* on page 2-85	Master	Flush of L2 memory system complete.
l2flushreq	*Signal* on page 2-85	Slave	Request flush of L2 memory system.
l2reset	*Signal* on page 2-85	Slave	Level2 reset.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-111 Ports (continued)

Name	Protocol	Type	Description
memorymapped_debug_s	PVBus	Slave	External debug interface.
niden[4]	*Signal* on page 2-85	Slave	External debug interface.
periphbase	*Value_64* on page 2-85	Slave	This port sets the base address of private peripheral region.
pmuirq[4]	*Signal* on page 2-85	Master	Interrupt signal from performance monitoring unit.
presetdbg	*Signal* on page 2-85	Slave	Debug reset.
pvbus_m0	PVBus	Master	The core will generate bus requests on this port.
rei[4]	*Signal* on page 2-85	Slave	Per core RAM Error Interrupt
reset[4]	*Signal* on page 2-85	Slave	Reset.
romaddr	*Value_64* on page 2-85	Slave	Debug ROM base address.
romaddrv	*Signal* on page 2-85	Slave	Debug ROM base address valid.
rvbaraddr[4]	*Value_64* on page 2-85	Slave	Reset vector base address.
sei[4]	*Signal* on page 2-85	Slave	Per core System Error physical pins.
smpen[4]	*Signal* on page 2-85	Master	This signals AMP or SMP mode for each core.
spiden[4]	*Signal* on page 2-85	Slave	External debug interface.
spniden[4]	*Signal* on page 2-85	Slave	External debug interface.
standbywfe[4]	*Signal* on page 2-85	Master	This signal indicates if a core is in WFE state.
standbywfi[4]	*Signal* on page 2-85	Master	This signal indicates if a core is in WFI state
standbywfil2	*Signal* on page 2-85	Master	This signal indicated all cores and L2 are idles and in low power state.
ticks[4]	*InstructionCount* on page 2-84	Master	This port should be connected to one of the two ticks ports on a 'visualisation' component, in order to display a running instruction count.
vcpumntirq[4]	*Signal* on page 2-85	Master	Interrupt signal for virtual CPU maintenance IRQ.
vfiq[4]	*Signal* on page 2-85	Slave	Virtualised FIQ.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-111 Ports (continued)

Name	Protocol	Type	Description
vinithi[4]	*Signal* on page 2-85	Slave	This signal controls of the location of the exception vectors at reset.
virq[4]	*Signal* on page 2-85	Slave	Virtualised IRQ.
virtio_s	PVBus	Slave	The virtio coherent port, hooks directly into the L2 system and becomes coherent (assuming attributes are set correctly).
vsei[4]	*Signal* on page 2-85	Slave	Per core virtual System Error physical pins.

Table 3-112 Parameters for ARM_Cortex-A35

Name	Type	Default value	Description
cpu0.AA64nAA32	bool	0x1	Register width configuration at reset. 0, AArch32. 1, AArch64.
cpu0.CFGEND	bool	0x0	Endianness configuration at reset. 0, little endian. 1, big endian.
cpu0.CFGTE	bool	0x0	Instruction set state when resetting into AArch32. 0, A32. 1, T32.
cpu0.CP15SDISABLE	bool	0x0	Initialize to disable access to some CP15 registers
cpu0.CP15SDISABLE2	bool	0x0	Initialize to disable access to some CP15 registers
cpu0.CRYPTODISABLE	bool	0x0	Disable cryptographic features.
cpu0.RVBARADDR	int	0x0	Value of RVBAR_ELx register.
cpu0.VINITHI	bool	0x0	Reset value of SCTLR.V.
cpu0.enable_trace_special_hlt_imm16	bool	0x0	Enable usage of parameter trace_special_hlt_imm16
cpu0.max_code_cache_mb	int	0x100	Maximum size of the simulation code cache (MiB). For platforms with more than 2 cores this limit will be scaled down. (e.g 1/8 for 16 or more cores)
cpu0.min_sync_level	int	0x0	Force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)
cpu0.semihosting-A32_HLT	int	0xf000	A32 HLT number for semihosting calls.
cpu0.semihosting-A64_HLT	int	0xf000	A64 HLT number for semihosting calls.
cpu0.semihosting-ARM_SVC	int	0x123456	A32 SVC number for semihosting calls.
cpu0.semihosting-T32_HLT	int	0x3c	T32 HLT number for semihosting calls.
cpu0.semihosting-Thumb_SVC	int	0xab	T32 SVC number for semihosting calls.
cpu0.semihosting-cmd_line	string	""	Command line available to semihosting calls.
cpu0.semihosting-cwd	string	""	Base directory for semihosting file access.
cpu0.semihosting-enable	bool	0x1	Enable semihosting SVC/HLT traps.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-112 Parameters for ARM_Cortex-A35 (continued)

Name	Type	Default value	Description
cpu0.semihosting-heap_base	int	0x0	Virtual address of heap base.
cpu0.semihosting-heap_limit	int	0xf000000	Virtual address of top of heap.
cpu0.semihosting-stack_base	int	0x10000000	Virtual address of base of descending stack.
cpu0.semihosting-stack_limit	int	0xf000000	Virtual address of stack limit.
cpu0.trace_special_hlt_imm16	int	0xf000	For this HLT number, IF enable_trace_special_hlt_imm16=true, skip performing usual HLT execution but call MTI trace if registered
cpu0.vfp-enable_at_reset	bool	0x0	Enable VFP in CPACR, CPPWR, NSACR at reset. Warning: Arm recommends going through the implementation's suggested VFP power-up sequence!
cpu0.vfp-present	bool	0x1	Set whether the model has VFP support

Table 3-113 Parameters for Cluster_ARM_Cortex-A35

Name	Type	Default value	Description
BROADCASTCACHEMAINT	bool	0x1	Enable broadcasting of cache maintenance operations to downstream caches. The broadcastcachemaint signal will override this value if used.
BROADCASTINNER	bool	0x1	Enable broadcasting of Inner Shareable transactions. The broadcastinner signal will override this value if used.
BROADCASTOUTER	bool	0x1	Enable broadcasting of Outer Shareable transactions. The broadcastouter signal will override this value if used.
CLUSTER_ID	int	0x0	Processor cluster ID value
DBGROMADDR	int	0x0	Initialization value of DBGDRAR register. Bits[47:12] of this register specify the ROM table physical address.
DBGROMADDRV	bool	0x0	If true, set bits[1:0] of the CP15 DBGDRAR to indicate that the address is valid
GICDISABLE	bool	0x1	Disable the new style GICv3 CPU interface in each core model. Should be left enabled unless the platform contains a GICv3 distributor.
PERIPHBASE	int	0x13080000	Base address of peripheral memory space
bus_type	int	0x0	Cosmetic change that changes reset value of L2ACTLR register. 0, ACE. 1, CHI. 2, AXI
cpi_div	int	0x1	Divider for calculating CPI (Cycles Per Instruction)
cpi_mul	int	0x1	Multiplier for calculating CPI (Cycles Per Instruction)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-113 Parameters for Cluster_ARM_Cortex-A35 (continued)

Name	Type	Default value	Description
dcache-hit_latency	int	0x0	L1 D-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
dcache-maintenance_latency	int	0x0	L1 D-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
dcache-miss_latency	int	0x0	L1 D-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
dcache-prefetch_enabled	bool	0x0	Enable simulation of data cache prefetching. This is only used when dcache-state_modelled=true
dcache-read_access_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per access (of size dcache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
dcache-read_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per byte accessed.dcache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
dcache-size	int	0x8000	L1 D-Cache size in bytes.
dcache-snoop_data_transfer_latency	int	0x0	L1 D-Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
dcache-state_modelled	bool	0x0	Set whether D-cache has stateful implementation
dcache-write_access_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per access (of size dcache-write_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-write_latency is set. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-113 Parameters for Cluster_ARM_Cortex-A35 (continued)

Name	Type	Default value	Description
dcache-write_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per byte accessed. dcache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
enable_simulation_performance_optimizations	bool	0x1	With this option enabled, the model will run more quickly, but be less accurate to exact CPU behavior. The model will still be functionally accurate for software, but may increase differences seen between hardware behavior and model behavior for certain workloads (it changes the micro-architectural value of stage12_tlb_size parameter to 1024).
icache-hit_latency	int	0x0	L1 I-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when icache-state_modelled=true.
icache-maintenance_latency	int	0x0	L1 I-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when icache-state_modelled=true.
icache-miss_latency	int	0x0	L1 I-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when icache-state_modelled=true.
icache-prefetch_enabled	bool	0x0	Enable simulation of instruction cache prefetching. This is only used when icache-state_modelled=true.
icache-read_access_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per access (of size icache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if icache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when icache-state_modelled=true.
icache-read_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per byte accessed.icache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when icache-state_modelled=true.
icache-size	int	0x8000	L1 I-Cache size in bytes.
icache-state_modelled	bool	0x0	Set whether I-cache has stateful implementation

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-113 Parameters for Cluster_ARM_Cortex-A35 (continued)

Name	Type	Default value	Description
l2cache-hit_latency	int	0x0	L2 Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
l2cache-maintenance_latency	int	0x0	L2 Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
l2cache-miss_latency	int	0x0	L2 Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
l2cache-read_access_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
l2cache-read_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per byte accessed.l2cache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
l2cache-size	int	0x80000	L2 Cache size in bytes.
l2cache-snoop_data_transfer_latency	int	0x0	L2 Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
l2cache-snoop_issue_latency	int	0x0	L2 Cache timing annotation latency for snoop accesses issued by this cache in total ticks. This is only used when dcache-state_modelled=true.
l2cache-write_access_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-write_latency is set. This is only used when dcache-state_modelled=true.
l2cache-write_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per byte accessed. l2cache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-113 Parameters for Cluster_ARM_Cortex-A35 (continued)

Name	Type	Default value	Description
`ptw_latency`	`int`	`0x0`	Page table walker latency for TA (Timing Annotation), expressed in simulation ticks
`tlb_latency`	`int`	`0x0`	TLB latency for TA (Timing Annotation), expressed in simulation ticks
`walk_cache_latency`	`int`	`0x0`	Walk cache latency for TA (Timing Annotation), expressed in simulation ticks

3.4.15 ARMCortexA53x1CT

ARMCortexA53x1CT CPU component. This model is written in C++ and models version r0p1 of the RTL.

ARMCortexA53x1CT contains the following CADI targets:

- ARM_Cortex-A53
- Cluster_ARM_Cortex-A53
- PVCache
- TlbCadi

ARMCortexA53x1CT contains the following MTI components:

- *4.11 ARM_Cortex-A53 - trace* on page 4-1152
- *4.48 ARMv8Cluster - trace* on page 4-2401
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402
- *4.68 DSU - trace* on page 4-2463
- *4.73 GICv3CPUInterfaceDecoder - trace* on page 4-2489
- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.108 PVCache - trace* on page 4-2761
- *4.114 TLB - trace* on page 4-2834

About ARMCortexA53x1CT

- The following components also exist:
  - ARMCortexA53x2CT.
  - ARMCortexA53x3CT.
  - ARMCortexA53x4CT.

  The per-core parameters are preceded by `cpu`*n*`.`, where *n* identifies the core (0-3).
- This model supports the Armv8-A Cryptography Extensions, which requires the Crypto plug-in to be loaded. See *Crypto plug-in* on page 5-2909 for details.
- The system designer decides whether a debug APB ties the external debug view with other system views. In the model, use a `PVBusDecoder` to direct traffic to the correct port, `dev_debug_s` or `memorymapped_debug_s`.

Differences between the CT model and RTL implementations

This component has the following differences from the corresponding revision of the RTL implementation:

- The value of the AArch64 PMCEID0_EL0 register, and the AArch32 alias of this register, differs in the model from the TRM value. The model value reflects the model counters.
- The mechanisms for setting the affinity fields of the MPIDR. The RTL has two ports:
  - CLUSTERIDAFF1[7:0].
  - CLUSTERIDAFF2[7:0].

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

AFF1 sets the value of MPIDR bits[15:8] and AFF2 sets the value of MPIDR bits[23:16]. In contrast, the model has a single `CLUSTER_ID` port. This difference allows the setting of bits[23:8] of the MPIDR using bits[15:0] of the `CLUSTER_ID` value.

- The memory mapped debug registers have a view for cores and a view for external debug agents. In the model, these views require two PVBus ports. In hardware, the system designer decides how the implementation differentiates the views.
- In the model, a single `peer` event port combines the functionality of the `eventi` and `evento` signals in the RTL.
- The Generic Timers are *Programmer's View* (PV) level abstractions: a model-specific protocol connects the `cntvalueb` port to the MemoryMappedCounterModule.
- The GIC CPU Interface is a PV level abstraction: a model-specific protocol connects the GIC CPU Interface to the GIC Distributor.
- The CoreSight™ *Cross Trigger Interface* (CTI) is a PV level abstraction: the interface is a model specific one.
- The model has no mechanism to read the internal memory that the Cache and TLB structures use, through the implementation defined region of the system coprocessor interface. This memory includes the RAM Index Register, IL1DATA Registers, DL1DATA Registers, and associated functionality.
- The model does not implement:
  - ETM registers.
  - The PMUEVENT bus.
  - The WARMRESETREQ signal. However, the warm reset code sequence (see the section *Code sequence to request a Warm reset as a result of RMR_ELx.RR* in the *Arm®v8-A Architecture Reference Manual*) makes the model simulate a warm reset of the core.
  - The PMUSNAPSHOTREQ and PMUSNAPSHOTACK signals.
  - The EXTERRIRQ and INTERRIRQ signals.
  - Processor dynamic-retention signals.
  - The SYSBARDISABLE signal.
  - The DBGPWRDUP, DBGPWRUPREQ, DBGNOPWRDWN, and DBGRSTREQ debug power management signals.
  - The RTL synthesis option to remove FP and ASE.
  - The RTL synthesis option for a Cortex-A15 style debug memory map.
  - Although NEON support is optional for the Cortex-A53 processor, this model does not implement the `ase-present` parameter. This means it is not possible to configure the model to not support NEON.
  - ECC and parity schemes are hardware-specific so are not supported.

Table 3-114 Ports

Name	Protocol	Type	Description
`CNTHPIRQ[4]`	*Signal* on page 2-85	Master	Timer signals to SoC.
`CNTPNSIRQ[4]`	*Signal* on page 2-85	Master	Timer signals to SoC.
`CNTPSIRQ[4]`	*Signal* on page 2-85	Master	Timer signals to SoC.
`CNTVIRQ[4]`	*Signal* on page 2-85	Master	Timer signals to SoC.
`aa64naa32[4]`	*Signal* on page 2-85	Slave	Register width after reset.
`acp_s`	PVBus	Slave	AXI ACP slave port.
`broadcastcachemaint`	*Signal* on page 2-85	Slave	Enable broadcasting of cache maintenance operations to downstream caches.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-114 Ports (continued)

Name	Protocol	Type	Description
broadcastinner	*Signal* on page 2-85	Slave	Enable broadcasting of Inner Shareable transactions.
broadcastouter	*Signal* on page 2-85	Slave	Enable broadcasting of Outer Shareable transactions.
cfgend[4]	*Signal* on page 2-85	Slave	This signal if for EE bit initialisation.
cfgsdisable	*Signal* on page 2-85	Slave	This signal disables write access to some secure Interrupt Controller registers.
cfgte[4]	*Signal* on page 2-85	Slave	This signal provides default exception handling state.
clk_in	ClockSignal	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the core executes instructions.
clrexmonack	*Signal* on page 2-85	Master	Acknowledge handshake signal for the clrexmonreq signal
clrexmonreq	*Signal* on page 2-85	Slave	Signals the clearing of an external global exclusive monitor
clusterid	*Value* on page 2-85	Slave	The port reads the value in CPU ID register field, bits[11:8] of the MPIDR.
cntvalueb	*CounterInterface* on page 2-84	Slave	Interface to SoC level counter module.
commirq[4]	*Signal* on page 2-85	Master	Interrupt signal from debug communications channel.
commrx[4]	*Signal* on page 2-85	Master	Receive portion of Data Transfer Register full.
commtx[4]	*Signal* on page 2-85	Master	Transmit portion of Data Transfer Register empty.
cp15sdisable[4]	*Signal* on page 2-85	Slave	This signal disables write access to some system control processor registers.
cpuporeset[4]	*Signal* on page 2-85	Slave	Power on reset. Initializes all the processor logic, including debug logic.
cryptodisable[4]	*Signal* on page 2-85	Slave	Disable cryptography extensions after reset.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-114 Ports (continued)

Name	Protocol	Type	Description
cti[4]	*v8EmbeddedCrossTrigger_controlprotocol* on page 2-84	Master	Cross trigger matrix port.
ctidbgirq[4]	*Signal* on page 2-85	Master	Cross Trigger Interface (CTI) interrupt trigger output.
dbgack[4]	*Signal* on page 2-85	Master	External debug interface.
dbgen[4]	*Signal* on page 2-85	Slave	External debug interface.
dbgl1rstdisable	*Signal* on page 2-85	Slave	Control ram clear on reset
dbgnopwrdwn[4]	*Signal* on page 2-85	Master	This signals relate to core power down.
dbgpwrupreq[4]	*Signal* on page 2-85	Master	This signals relate to core power down.
dev_debug_s	PVBus	Slave	External debug interface.
edbgrq[4]	*Signal* on page 2-85	Slave	External debug interface.
event	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE.
fiq[4]	*Signal* on page 2-85	Slave	This signal drives the CPUs fast-interrupt handling.
gicv3_redistributor_s[4]	*GICv3Comms* on page 2-84	Slave	GICv3 AXI-stream port.
irq[4]	*Signal* on page 2-85	Slave	This signal drives the CPUs interrupt handling.
l2flushdone	*Signal* on page 2-85	Master	Flush of L2 memory system complete.
l2flushreq	*Signal* on page 2-85	Slave	Request flush of L2 memory system.
l2reset	*Signal* on page 2-85	Slave	Reset the shared L2 memory system controller.
l2rstdisable	*Signal* on page 2-85	Slave	-
memorymapped_debug_s	PVBus	Slave	External debug interface.
niden[4]	*Signal* on page 2-85	Slave	External debug interface.
periphbase	*Value_64* on page 2-85	Slave	This port sets the base address of private peripheral region.
pmuirq[4]	*Signal* on page 2-85	Master	Interrupt signal from performance monitoring unit.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-114 Ports (continued)

Name	Protocol	Type	Description
presetdbg	*Signal* on page 2-85	Slave	Initialize the shared debug APB, Cross Trigger Interface (CTI), and Cross Trigger Matrix (CTM) logic.
pvbus_m0	PVBus	Master	The core will generate bus requests on this port.
rei[4]	*Signal* on page 2-85	Slave	Individual processor RAM Error Interrupt signal input.
reset[4]	*Signal* on page 2-85	Slave	Raising this signal will put the core into reset mode.
romaddr	*Value_64* on page 2-85	Slave	Debug ROM base address.
romaddrv	*Signal* on page 2-85	Slave	Debug ROM base address valid.
rvbaraddr[4]	*Value_64* on page 2-85	Slave	Reset vector base address.
sei[4]	*Signal* on page 2-85	Slave	Per core System Error physical pins.
smpen[4]	*Signal* on page 2-85	Master	This signals AMP or SMP mode for each core.
spiden[4]	*Signal* on page 2-85	Slave	External debug interface.
spniden[4]	*Signal* on page 2-85	Slave	External debug interface.
standbywfe[4]	*Signal* on page 2-85	Master	This signal indicates if a core is in WFE state.
standbywfi[4]	*Signal* on page 2-85	Master	This signal indicates if a core is in WFI state.
standbywfil2	*Signal* on page 2-85	Master	Indicate that all the individual processors and the L2 systems are in a WFI state.
ticks[4]	*InstructionCount* on page 2-84	Master	This port should be connected to one of the two ticks ports on a 'visualisation' component, in order to display a running instruction count.
vcpumntirq[4]	*Signal* on page 2-85	Master	Interrupt signal for virtual CPU maintenance IRQ.
vfiq[4]	*Signal* on page 2-85	Slave	Virtual FIQ.
vinithi[4]	*Signal* on page 2-85	Slave	This signal controls of the location of the exception vectors at reset.
virq[4]	*Signal* on page 2-85	Slave	Virtual IRQ.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-114 Ports (continued)

Name	Protocol	Type	Description
virtio_s	PVBus	Slave	The virtio coherent port, hooks directly into the L2 system and becomes coherent (assuming attributes are set correctly).
vsei[4]	*Signal* on page 2-85	Slave	Processor Virtual System Error Interrupt request.

Table 3-115 Parameters for Cluster_ARM_Cortex-A53

Name	Type	Default value	Description
BROADCASTCACHEMAINT	bool	0x1	Enable broadcasting of cache maintenance operations to downstream caches. The broadcastcachemaint signal will override this value if used.
BROADCASTINNER	bool	0x1	Enable broadcasting of Inner Shareable transactions. The broadcastinner signal will override this value if used.
BROADCASTOUTER	bool	0x1	Enable broadcasting of Outer Shareable transactions. The broadcastouter signal will override this value if used.
CLUSTER_ID	int	0x0	Processor cluster ID value
DBGROMADDR	int	0x0	Initialization value of DBGDRAR register. Bits[47:12] of this register specify the ROM table physical address.
DBGROMADDRV	bool	0x0	If true, set bits[1:0] of the CP15 DBGDRAR to indicate that the address is valid
GICDISABLE	bool	0x1	Disable the new style GICv3 CPU interface in each core model. Should be left enabled unless the platform contains a GICv3 distributor.
PERIPHBASE	int	0x13080000	Base address of peripheral memory space
bus_type	int	0x0	Cosmetic change that changes reset value of L2ACTLR register. 0, ACE. 1, CHI. 2, AXI
cpi_div	int	0x1	Divider for calculating CPI (Cycles Per Instruction)
cpi_mul	int	0x1	Multiplier for calculating CPI (Cycles Per Instruction)
dcache-hit_latency	int	0x0	L1 D-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
dcache-maintenance_latency	int	0x0	L1 D-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-115 Parameters for Cluster_ARM_Cortex-A53 (continued)

Name	Type	Default value	Description
dcache-miss_latency	int	0x0	L1 D-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
dcache-prefetch_enabled	bool	0x0	Enable simulation of data cache prefetching. This is only used when dcache-state_modelled=true
dcache-read_access_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per access (of size dcache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
dcache-read_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per byte accessed.dcache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
dcache-size	int	0x8000	L1 D-Cache size in bytes.
dcache-snoop_data_transfer_latency	int	0x0	L1 D-Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
dcache-state_modelled	bool	0x0	Set whether D-cache has stateful implementation
dcache-write_access_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per access (of size dcache-write_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-write_latency is set. This is only used when dcache-state_modelled=true.
dcache-write_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per byte accessed. dcache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-115 Parameters for Cluster_ARM_Cortex-A53 (continued)

Name	Type	Default value	Description
enable_simulation_performance_optimizations	bool	0x1	With this option enabled, the model will run more quickly, but be less accurate to exact CPU behavior. The model will still be functionally accurate for software, but may increase differences seen between hardware behavior and model behavior for certain workloads (it changes the micro-architectural value of stage12_tlb_size parameter to 1024).
icache-hit_latency	int	0x0	L1 I-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when icache-state_modelled=true.
icache-maintenance_latency	int	0x0	L1 I-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when icache-state_modelled=true.
icache-miss_latency	int	0x0	L1 I-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when icache-state_modelled=true.
icache-prefetch_enabled	bool	0x0	Enable simulation of instruction cache prefetching. This is only used when icache-state_modelled=true.
icache-read_access_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per access (of size icache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if icache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when icache-state_modelled=true.
icache-read_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per byte accessed.icache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when icache-state_modelled=true.
icache-size	int	0x8000	L1 I-Cache size in bytes.
icache-state_modelled	bool	0x0	Set whether I-cache has stateful implementation
l2cache-hit_latency	int	0x0	L2 Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
l2cache-maintenance_latency	int	0x0	L2 Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-115 Parameters for Cluster_ARM_Cortex-A53 (continued)

Name	Type	Default value	Description
l2cache-miss_latency	int	0x0	L2 Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
l2cache-read_access_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
l2cache-read_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per byte accessed.l2cache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
l2cache-size	int	0x80000	L2 Cache size in bytes.
l2cache-snoop_data_transfer_latency	int	0x0	L2 Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
l2cache-snoop_issue_latency	int	0x0	L2 Cache timing annotation latency for snoop accesses issued by this cache in total ticks. This is only used when dcache-state_modelled=true.
l2cache-write_access_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-write_latency is set. This is only used when dcache-state_modelled=true.
l2cache-write_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per byte accessed. l2cache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
patch_level	int	0x1	Cosmetic change to patch number in MIDR/ MIDR_EL1. Corresponds to the Y in rXpY.
ptw_latency	int	0x0	Page table walker latency for TA (Timing Annotation), expressed in simulation ticks
revision_number	int	0x0	Cosmetic change to revision number in MIDR/ MIDR_EL1. Corresponds to the X in rXpY.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-115 Parameters for Cluster_ARM_Cortex-A53 (continued)

Name	Type	Default value	Description
tlb_latency	int	0x0	TLB latency for TA (Timing Annotation), expressed in simulation ticks
walk_cache_latency	int	0x0	Walk cache latency for TA (Timing Annotation), expressed in simulation ticks

Table 3-116 Parameters for ARM_Cortex-A53

Name	Type	Default value	Description
cpu0.AA64nAA32	bool	0x1	Register width configuration at reset. 0, AArch32. 1, AArch64.
cpu0.CFGEND	bool	0x0	Endianness configuration at reset. 0, little endian. 1, big endian.
cpu0.CFGTE	bool	0x0	Instruction set state when resetting into AArch32. 0, A32. 1, T32.
cpu0.CP15SDISABLE	bool	0x0	Initialize to disable access to some CP15 registers
cpu0.CRYPTODISABLE	bool	0x0	Disable cryptographic features.
cpu0.RVBARADDR	int	0x0	Value of RVBAR_ELx register.
cpu0.VINITHI	bool	0x0	Reset value of SCTLR.V.
cpu0.enable_trace_special_hlt_imm16	bool	0x0	Enable usage of parameter trace_special_hlt_imm16
cpu0.max_code_cache_mb	int	0x100	Maximum size of the simulation code cache (MiB). For platforms with more than 2 cores this limit will be scaled down. (e.g 1/8 for 16 or more cores)
cpu0.min_sync_level	int	0x0	Force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)
cpu0.semihosting-A32_HLT	int	0xf000	A32 HLT number for semihosting calls.
cpu0.semihosting-A64_HLT	int	0xf000	A64 HLT number for semihosting calls.
cpu0.semihosting-ARM_SVC	int	0x123456	A32 SVC number for semihosting calls.
cpu0.semihosting-T32_HLT	int	0x3c	T32 HLT number for semihosting calls.
cpu0.semihosting-Thumb_SVC	int	0xab	T32 SVC number for semihosting calls.
cpu0.semihosting-cmd_line	string	""	Command line available to semihosting calls.
cpu0.semihosting-cwd	string	""	Base directory for semihosting file access.
cpu0.semihosting-enable	bool	0x1	Enable semihosting SVC/HLT traps.
cpu0.semihosting-heap_base	int	0x0	Virtual address of heap base.
cpu0.semihosting-heap_limit	int	0xf000000	Virtual address of top of heap.
cpu0.semihosting-stack_base	int	0x10000000	Virtual address of base of descending stack.
cpu0.semihosting-stack_limit	int	0xf000000	Virtual address of stack limit.
cpu0.trace_special_hlt_imm16	int	0xf000	For this HLT number, IF enable_trace_special_hlt_imm16=true, skip performing usual HLT execution but call MTI trace if registered

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-116 Parameters for ARM_Cortex-A53 (continued)

Name	Type	Default value	Description
cpu0.vfp-enable_at_reset	bool	0x0	Enable VFP in CPACR, CPPWR, NSACR at reset. Warning: Arm recommends going through the implementation's suggested VFP power-up sequence!
cpu0.vfp-present	bool	0x1	Set whether the model has VFP support

3.4.16 ARMCortexA55CT

ARMCortexA55CT CPU component. This model is written in C++ and models version r0p0 of the RTL.

ARMCortexA55CT contains the following CADI targets:

- ARM_Cortex-A55
- Cluster_ARM_Cortex-A55
- PVCache
- TlbCadi

ARMCortexA55CT contains the following MTI components:

- *4.12 ARM_Cortex-A55 - trace* on page 4-1190
- *4.48 ARMv8Cluster - trace* on page 4-2401
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402
- *4.68 DSU - trace* on page 4-2463
- *4.73 GICv3CPUInterfaceDecoder - trace* on page 4-2489
- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.108 PVCache - trace* on page 4-2761
- *4.114 TLB - trace* on page 4-2834

About ARMCortexA55CT

The model supports the following features:

- DynamIQ™ (FCM) system registers.
- Per-core L2 cache.
- A P-Channel for the cluster and for each core.
- Optional peripheral port.
- L3 cache partition.
- Per-core clock.
- BROADCASTPERSIST pin.

The model does not support the following features:

- BROADCASTCACHEMAINTPOU pin.
- COREINSTRRET, COREINSTRRUN, or nPMBIRQ signals.
- DynamIQ features that are irrelevant to the programmers' view simulation, for example:
  - Automatic CPU retention mode.
  - Level-3 Cache RAM retention.
- 256-bit wide output transactions.
- ECC and parity schemes are hardware-specific so are not supported.
- Self-test features (MBIST).
- Latency configuration.
- Snoop filtering.
- Debug power management signals DBGPWRDUP and DBGRSTREQ are not implemented, but DBGPWRUPREQ and DBGNOPWRDWN are implemented.
- Cache stashing capability.
- Dual ACE masters.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

This component has a variable number of cores per cluster, specified using the NUM_CORES parameter. The following components also exist, with a fixed number of cores per cluster:

- ARMCortexA55x1CT.
- ARMCortexA55x2CT.
- ARMCortexA55x3CT.
- ARMCortexA55x4CT.
- ARMCortexA55x8CT.

The per-core parameters are preceded by cpu*n*., where *n* identifies the core (0-7).

Table 3-117 Ports

Name	Protocol	Type	Description
AENDMP	*Value_64* on page 2-85	Slave	Port to obtain end address of valid peripheral address range (exclusive).
ASTARTMP	*Value_64* on page 2-85	Slave	Port to obtain start address of valid peripheral address range (inclusive).
CNTHPIRQ[8]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTHVIRQ[8]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTPNSIRQ[8]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTPSIRQ[8]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTVIRQ[8]	*Signal* on page 2-85	Master	Timer signals to SOC.
aa64naa32[8]	*Signal* on page 2-85	Slave	Register width after reset.
acp_s	PVBus	Slave	AXI ACP slave port.
broadcastatomic	*Signal* on page 2-85	Slave	CHI defined pins.
broadcastcachemaint	*Signal* on page 2-85	Slave	ACE defined pins.
broadcastouter	*Signal* on page 2-85	Slave	ACE defined pins.
broadcastpersist	*Signal* on page 2-85	Slave	CHI defined pins.
cfgend[8]	*Signal* on page 2-85	Slave	This signal if for EE bit initialisation.
cfgsdisable	*Signal* on page 2-85	Slave	This signal disables write access to some secure Interrupt Controller registers.
cfgte[8]	*Signal* on page 2-85	Slave	This signal provides default exception handling state.
clk_in	ClockSignal	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the cluster level components run e.g cluster level timers, caches and pmu.
clusterid	*Value* on page 2-85	Slave	The port reads the value in CPU ID register field, bits[11:8] of the MPIDR.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-117 Ports (continued)

Name	Protocol	Type	Description
clusterpmuirq	*Signal* on page 2-85	Master	DynamIQ pmu irq
cntvalueb	*CounterInterface* on page 2-84	Slave	Interface to SoC level counter module.
commirq[8]	*Signal* on page 2-85	Master	Interrupt signal from debug communications channel.
core_clk_in[8]	ClockSignal	Slave	The clock signal connected to the core_clk_in port is used to determine the rate at which each core executes instructions.
cp15sdisable[8]	*Signal* on page 2-85	Slave	This signal disables write access to some system control processor registers.
cpuporeset[8]	*Signal* on page 2-85	Slave	CPU power on reset. Initializes all the processor logic, including debug logic.
cryptodisable[8]	*Signal* on page 2-85	Slave	Disable cryptography extensions after reset.
cti[8]	*v8EmbeddedCrossTrigger_controlprotocol* on page 2-84	Master	Cross trigger matrix port.
ctidbgirq[8]	*Signal* on page 2-85	Master	Cross Trigger Interface (CTI) interrupt trigger output.
dbgen[8]	*Signal* on page 2-85	Slave	External debug interface.
dbgnopwrdwn[8]	*Signal* on page 2-85	Master	No power-down request.
dbgpwrupreq[8]	*Signal* on page 2-85	Master	Debug power up request.
dev_debug_s	PVBus	Slave	External debug interface.
event	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE.
fiq[8]	*Signal* on page 2-85	Slave	This signal drives the CPUs fast-interrupt handling.
gicv3_redistributor_s[8]	*GICv3Comms* on page 2-84	Slave	GICv3 AXI-stream port.
irq[8]	*Signal* on page 2-85	Slave	This signal drives the CPUs interrupt handling.
memorymapped_debug_s	PVBus	Slave	External debug interface.
niden[8]	*Signal* on page 2-85	Slave	External debug interface.
pchannel_cluster	*PChannel* on page 2-80	Slave	PChannel for cluster.
pchannel_core[8]	*PChannel* on page 2-80	Slave	PChannels for cores

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-117 Ports (continued)

Name	Protocol	Type	Description
pmuirq[8]	*Signal* on page 2-85	Master	Interrupt signal from performance monitoring unit.
presetdbg	*Signal* on page 2-85	Slave	Initialize the shared debug APB, Cross Trigger Interface (CTI), and Cross Trigger Matrix (CTM) logic.
pvbus_m0	PVBus	Master	The core will generate bus requests on this port.
pvbus_periph_m	PVBus	Master	The core can generate peripheral bus request on this port.
rei[8]	*Signal* on page 2-85	Slave	Per core RAM Error Interrupt.
reset[8]	*Signal* on page 2-85	Slave	Raising this signal will put the core into reset mode.
rvbaraddr[8]	*Value_64* on page 2-85	Slave	Reset vector base address.
sci_m	SystemCoherencyInterface	Master	System coherency interface port, which is used to take the whole cluster into/out-of coherency domain
sei[8]	*Signal* on page 2-85	Slave	Per core System Error physical pins.
spiden[8]	*Signal* on page 2-85	Slave	External debug interface.
spniden[8]	*Signal* on page 2-85	Slave	External debug interface.
sporeset	*Signal* on page 2-85	Slave	A single cluster-wide power on reset signal for all resettable registers in DynamIQ.
ticks[8]	*InstructionCount* on page 2-84	Master	This port should be connected to one of the two ticks ports on a 'visualisation' component, in order to display a running instruction count.
vcpumntirq[8]	*Signal* on page 2-85	Master	Interrupt signal for virtual CPU maintenance IRQ.
vfiq[8]	*Signal* on page 2-85	Slave	Virtualised FIQ.
vinithi[8]	*Signal* on page 2-85	Slave	This signal controls of the location of the exception vectors at reset.
virq[8]	*Signal* on page 2-85	Slave	Virtualised IRQ.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-117 Ports (continued)

Name	Protocol	Type	Description
virtio_s	PVBus	Slave	The virtio coherent port, hooks directly into the L2 system and becomes coherent (assuming attributes are set correctly).
vsei[8]	*Signal* on page 2-85	Slave	Per core virtual System Error physical pins.

Table 3-118 Parameters for ARM_Cortex-A55

Name	Type	Default value	Description
cpu0.AA64nAA32	bool	0x1	Register width configuration at reset. 0, AArch32. 1, AArch64.
cpu0.CFGEND	bool	0x0	Endianness configuration at reset. 0, little endian. 1, big endian.
cpu0.CFGTE	bool	0x0	Instruction set state when resetting into AArch32. 0, A32. 1, T32.
cpu0.CP15SDISABLE	bool	0x0	Initialize to disable access to some CP15 registers
cpu0.CRYPTODISABLE	bool	0x0	Disable cryptographic features.
cpu0.RVBARADDR	int	0x0	Value of RVBAR_ELx register.
cpu0.VINITHI	bool	0x0	Reset value of SCTLR.V.
cpu0.enable_trace_special_hlt_imm16	bool	0x0	Enable usage of parameter trace_special_hlt_imm16
cpu0.l2cache-hit_latency	int	0x0	L2 Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
cpu0.l2cache-maintenance_latency	int	0x0	L2 Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
cpu0.l2cache-miss_latency	int	0x0	L2 Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
cpu0.l2cache-read_access_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-118 Parameters for ARM_Cortex-A55 (continued)

Name	Type	Default value	Description
cpu0.l2cache-read_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per byte accessed.l2cache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
cpu0.l2cache-size	int	0x80000	L2 Cache size in bytes.
cpu0.l2cache-snoop_data_transfer_latency	int	0x0	L2 Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
cpu0.l2cache-snoop_issue_latency	int	0x0	L2 Cache timing annotation latency for snoop accesses issued by this cache in total ticks. This is only used when dcache-state_modelled=true.
cpu0.l2cache-write_access_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-write_latency is set. This is only used when dcache-state_modelled=true.
cpu0.l2cache-write_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per byte accessed. l2cache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
cpu0.max_code_cache_mb	int	0x100	Maximum size of the simulation code cache (MiB). For platforms with more than 2 cores this limit will be scaled down. (e.g 1/8 for 16 or more cores)
cpu0.min_sync_level	int	0x0	Force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)
cpu0.semihosting-A32_HLT	int	0xf000	A32 HLT number for semihosting calls.
cpu0.semihosting-A64_HLT	int	0xf000	A64 HLT number for semihosting calls.
cpu0.semihosting-ARM_SVC	int	0x123456	A32 SVC number for semihosting calls.
cpu0.semihosting-T32_HLT	int	0x3c	T32 HLT number for semihosting calls.
cpu0.semihosting-Thumb_SVC	int	0xab	T32 SVC number for semihosting calls.
cpu0.semihosting-cmd_line	string	""	Command line available to semihosting calls.
cpu0.semihosting-cwd	string	""	Base directory for semihosting file access.
cpu0.semihosting-enable	bool	0x1	Enable semihosting SVC/HLT traps.
cpu0.semihosting-heap_base	int	0x0	Virtual address of heap base.
cpu0.semihosting-heap_limit	int	0xf000000	Virtual address of top of heap.
cpu0.semihosting-stack_base	int	0x10000000	Virtual address of base of descending stack.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-118 Parameters for ARM_Cortex-A55 (continued)

Name	Type	Default value	Description
cpu0.semihosting-stack_limit	int	0xf000000	Virtual address of stack limit.
cpu0.trace_special_hlt_imm16	int	0xf000	For this HLT number, IF enable_trace_special_hlt_imm16=true, skip performing usual HLT execution but call MTI trace if registered
cpu0.vfp-enable_at_reset	bool	0x0	Enable VFP in CPACR, CPPWR, NSACR at reset. Warning: Arm recommends going through the implementation's suggested VFP power-up sequence!
cpu0.vfp-present	bool	0x1	Set whether the model has VFP support

Table 3-119 Parameters for Cluster_ARM_Cortex-A55

Name	Type	Default value	Description
BROADCASTATOMIC	bool	0x1	Enable broadcasting of atomic operation. The broadcastatomic signal will override this value if used
BROADCASTCACHEMAINT	bool	0x1	Enable broadcasting of cache maintenance operations to downstream caches. The broadcastcachemaint signal will override this value if used.
BROADCASTOUTER	bool	0x1	Enable broadcasting of Outer Shareable transactions. The broadcastouter signal will override this value if used.
BROADCASTPERSIST	bool	0x1	Enable broadcasting of cache clean to the point of persistence operations. The broadcastpersist signal will override this value if used
CLUSTER_ID	int	0x0	Processor cluster ID value
GICDISABLE	bool	0x1	Disable the new style GICv3 CPU interface in each core model. Should be left enabled unless the platform contains a GICv3 distributor.
bus_type	int	0x0	Cosmetic change that changes reset value of L2ACTLR register. 0, ACE. 1, CHI. 2, AXI
cpi_div	int	0x1	Divider for calculating CPI (Cycles Per Instruction)
cpi_mul	int	0x1	Multiplier for calculating CPI (Cycles Per Instruction)
dcache-hit_latency	int	0x0	L1 D-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
dcache-maintenance_latency	int	0x0	L1 D-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-119 Parameters for Cluster_ARM_Cortex-A55 (continued)

Name	Type	Default value	Description
dcache-miss_latency	int	0x0	L1 D-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
dcache-prefetch_enabled	bool	0x0	Enable simulation of data cache prefetching. This is only used when dcache-state_modelled=true
dcache-read_access_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per access (of size dcache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
dcache-read_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per byte accessed.dcache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
dcache-size	int	0x8000	L1 D-Cache size in bytes.
dcache-snoop_data_transfer_latency	int	0x0	L1 D-Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
dcache-state_modelled	bool	0x0	Set whether D-cache has stateful implementation
dcache-write_access_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per access (of size dcache-write_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-write_latency is set. This is only used when dcache-state_modelled=true.
dcache-write_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per byte accessed. dcache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
enable_simulation_performance_optimizations	bool	0x1	With this option enabled, the model will run more quickly, but be less accurate to exact CPU behavior. The model will still be functionally accurate for software, but may increase differences seen between hardware behavior and model behavior for certain workloads (it changes the micro-architectural value of stage12_tlb_size parameter to 1024).

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-119 Parameters for Cluster_ARM_Cortex-A55 (continued)

Name	Type	Default value	Description
has_dot_product	int	0x2	Possible values of this parameter are: - 1, feature not implemented. - 2, feature is implemented.
has_peripheral_port	bool	0x0	If true, additional AXI peripheral port is configured.
icache-hit_latency	int	0x0	L1 I-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when icache-state_modelled=true.
icache-maintenance_latency	int	0x0	L1 I-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when icache-state_modelled=true.
icache-miss_latency	int	0x0	L1 I-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when icache-state_modelled=true.
icache-prefetch_enabled	bool	0x0	Enable simulation of instruction cache prefetching. This is only used when icache-state_modelled=true.
icache-read_access_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per access (of size icache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if icache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when icache-state_modelled=true.
icache-read_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per byte accessed.icache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when icache-state_modelled=true.
icache-size	int	0x8000	L1 I-Cache size in bytes.
icache-state_modelled	bool	0x0	Set whether I-cache has stateful implementation
l3cache-hit_latency	int	0x0	L3 Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
l3cache-maintenance_latency	int	0x0	L3 Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
l3cache-miss_latency	int	0x0	L3 Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-119 Parameters for Cluster_ARM_Cortex-A55 (continued)

Name	Type	Default value	Description
l3cache-read_access_latency	int	0x0	L3 Cache timing annotation latency for read accesses given in ticks per access (of size l3cache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l3cache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
l3cache-read_latency	int	0x0	L3 Cache timing annotation latency for read accesses given in ticks per byte accessed.l3cache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
l3cache-size	int	0x80000	L3 Cache size in bytes.
l3cache-snoop_data_transfer_latency	int	0x0	L3 Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
l3cache-snoop_issue_latency	int	0x0	L3 Cache timing annotation latency for snoop accesses issued by this cache in total ticks. This is only used when dcache-state_modelled=true.
l3cache-write_access_latency	int	0x0	L3 Cache timing annotation latency for write accesses given in ticks per access (of size l3cache-write_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l3cache-write_latency is set. This is only used when dcache-state_modelled=true.
l3cache-write_latency	int	0x0	L3 Cache timing annotation latency for write accesses given in ticks per byte accessed. l3cache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
pchannel_treat_simreset_as_poreset	bool	0x0	Register core as ON state to cluster with simulation reset.
periph_address_end	int	0x0	End address for peripheral port address range exclusive(corresponds to AENDMP input signal).
periph_address_start	int	0x0	Start address for peripheral port address range inclusive(corresponds to ASTARTMP input signal).
ptw_latency	int	0x0	Page table walker latency for TA (Timing Annotation), expressed in simulation ticks

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-119 Parameters for Cluster_ARM_Cortex-A55 (continued)

Name	Type	Default value	Description
`tlb_latency`	`int`	`0x0`	TLB latency for TA (Timing Annotation), expressed in simulation ticks
`walk_cache_latency`	`int`	`0x0`	Walk cache latency for TA (Timing Annotation), expressed in simulation ticks

3.4.17 ARMCortexA55CT_CortexA75CT

ARMCortexA55CT_CortexA75CT CPU component. This model is written in C++.

ARMCortexA55CT_CortexA75CT contains the following CADI targets:

- ARM_Cortex-A55
- Cluster_ARM_Cortex-A55_Cortex-A75
- Subcluster_ARM_Cortex-A55
- Subcluster_ARM_Cortex-A75

ARMCortexA55CT_CortexA75CT contains the following MTI components:

- *4.12 ARM_Cortex-A55 - trace* on page 4-1190
- *4.20 ARM_Cortex-A75 - trace* on page 4-1479
- *4.48 ARMv8Cluster - trace* on page 4-2401
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402
- *4.68 DSU - trace* on page 4-2463
- *4.73 GICv3CPUInterfaceDecoder - trace* on page 4-2489
- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.108 PVCache - trace* on page 4-2761
- *4.114 TLB - trace* on page 4-2834

About ARMCortexA55CT_CortexA75CT

For information about the cores in this model, see:

- *3.4.16 ARMCortexA55CT* on page 3-342.
- *3.4.25 ARMCortexA75CT* on page 3-434.

Table 3-120 Ports

Name	Protocol	Type	Description
`AENDMP`	*`Value_64` on page 2-85*	Slave	DynamIQ port to obtain end address of valid peripheral address range (exclusive).
`ASTARTMP`	*`Value_64` on page 2-85*	Slave	DynamIQ port to obtain start address of valid peripheral address range (inclusive).
`CNTHPIRQ[8]`	*`Signal` on page 2-85*	Master	Timer signals to SOC.
`CNTHVIRQ[8]`	*`Signal` on page 2-85*	Master	Timer signals to SOC.
`CNTPNSIRQ[8]`	*`Signal` on page 2-85*	Master	Timer signals to SOC.
`CNTPSIRQ[8]`	*`Signal` on page 2-85*	Master	Timer signals to SOC.
`CNTVIRQ[8]`	*`Signal` on page 2-85*	Master	Timer signals to SOC.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-120 Ports (continued)

Name	Protocol	Type	Description
aa64naa32[8]	*Signal* on page 2-85	Slave	Register width after reset.
acp_s	PVBus	Slave	AXI ACP slave port
broadcastatomic	*Signal* on page 2-85	Slave	CHI defined pins.
broadcastcachemaint	*Signal* on page 2-85	Slave	ACE defined pins.
broadcastouter	*Signal* on page 2-85	Slave	ACE defined pins.
broadcastpersist	*Signal* on page 2-85	Slave	CHI defined pins.
cfgend[8]	*Signal* on page 2-85	Slave	This signal if for EE bit initialisation.
cfgsdisable	*Signal* on page 2-85	Slave	This signal disables write access to some secure Interrupt Controller registers.
cfgte[8]	*Signal* on page 2-85	Slave	This signal provides default exception handling state.
clk_in	ClockSignal	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the cluster level components run e.g cluster level timers, caches and pmu.
clusterid	*Value* on page 2-85	Slave	The port reads the value in CPU ID register field, bits[11:8] of the MPIDR.
clusterpmuirq	*Signal* on page 2-85	Master	DynamIQ pmu irq
cntvalueb	*CounterInterface* on page 2-84	Slave	Interface to SoC level counter module.
commirq[8]	*Signal* on page 2-85	Master	Interrupt signal from debug communications channel.
core_clk_in[8]	ClockSignal	Slave	The clock signal connected to the core_clk_in port is used to determine the rate at which each core executes instructions.
cp15sdisable[8]	*Signal* on page 2-85	Slave	This signal disables write access to some system control processor registers.
cpuporeset[8]	*Signal* on page 2-85	Slave	CPU power on reset. Initializes all the processor logic, including debug logic.
cryptodisable[8]	*Signal* on page 2-85	Slave	Disable cryptography extensions after reset.
cti[8]	*v8EmbeddedCrossTrigger_controlprotocol* on page 2-84	Master	Cross trigger matrix port.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-120 Ports (continued)

Name	Protocol	Type	Description
ctidbgirq[8]	*Signal* on page 2-85	Master	Cross Trigger Interface (CTI) interrupt trigger output.
dbgen[8]	*Signal* on page 2-85	Slave	External debug interface.
dbgnopwrdwn[8]	*Signal* on page 2-85	Master	No power-down request.
dbgpwrupreq[8]	*Signal* on page 2-85	Master	Debug power up request.
dev_debug_s	PVBus	Slave	External debug interface.
event	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE.
fiq[8]	*Signal* on page 2-85	Slave	This signal drives the CPUs fast-interrupt handling.
gicv3_redistributor_s[8]	*GICv3Comms* on page 2-84	Slave	GICv3 AXI-stream port.
irq[8]	*Signal* on page 2-85	Slave	This signal drives the CPUs interrupt handling.
memorymapped_debug_s	PVBus	Slave	External debug interface.
niden[8]	*Signal* on page 2-85	Slave	External debug interface.
pchannel_cluster	*PChannel* on page 2-80	Slave	PChannel for cluster.
pchannel_core[8]	*PChannel* on page 2-80	Slave	PChannels for cores
pmuirq[8]	*Signal* on page 2-85	Master	Interrupt signal from performance monitoring unit.
presetdbg	*Signal* on page 2-85	Slave	Initialize the shared debug APB, Cross Trigger Interface (CTI), and Cross Trigger Matrix (CTM) logic.
pvbus_m0	PVBus	Master	The core will generate bus requests on this port.
pvbus_periph_m	PVBus	Master	The core can generate peripheral bus request on this port.
rei[8]	*Signal* on page 2-85	Slave	Per core RAM Error Interrupt.
reset[8]	*Signal* on page 2-85	Slave	Raising this signal will put the core into reset mode.
rvbaraddr[8]	*Value_64* on page 2-85	Slave	Reset vector base address.
sci_m	SystemCoherencyInterface	Master	System coherency interface port, which is used to take the whole cluster into/out-of coherency domain

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-120 Ports (continued)

Name	Protocol	Type	Description
sei[8]	*Signal* on page 2-85	Slave	Per core System Error physical pins.
spiden[8]	*Signal* on page 2-85	Slave	External debug interface.
spniden[8]	*Signal* on page 2-85	Slave	External debug interface.
sporeset	*Signal* on page 2-85	Slave	A single cluster-wide power on reset signal for all resettable registers in DynamIQ.
ticks[8]	*InstructionCount* on page 2-84	Master	This port should be connected to one of the two ticks ports on a 'visualisation' component, in order to display a running instruction count.
vcpumntirq[8]	*Signal* on page 2-85	Master	Interrupt signal for virtual CPU maintenance IRQ.
vfiq[8]	*Signal* on page 2-85	Slave	Virtualised FIQ.
vinithi[8]	*Signal* on page 2-85	Slave	This signal controls of the location of the exception vectors at reset.
virq[8]	*Signal* on page 2-85	Slave	Virtualised IRQ.
virtio_s	PVBus	Slave	The virtio coherent port, hooks directly into the L3 system and becomes coherent (assuming attributes are set correctly).
vsei[8]	*Signal* on page 2-85	Slave	Per core virtual System Error physical pins.

Table 3-121 Parameters for ARM_Cortex-A55

Name	Type	Default value	Description
cpu0.AA64nAA32	bool	0x1	Register width configuration at reset. 0, AArch32. 1, AArch64.
cpu0.CFGEND	bool	0x0	Endianness configuration at reset. 0, little endian. 1, big endian.
cpu0.CFGTE	bool	0x0	Instruction set state when resetting into AArch32. 0, A32. 1, T32.
cpu0.CP15SDISABLE	bool	0x0	Initialize to disable access to some CP15 registers
cpu0.CRYPTODISABLE	bool	0x0	Disable cryptographic features.
cpu0.RVBARADDR	int	0x0	Value of RVBAR_ELx register.
cpu0.VINITHI	bool	0x0	Reset value of SCTLR.V.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-121 Parameters for ARM_Cortex-A55 (continued)

Name	Type	Default value	Description
cpu0.enable_trace_special_hlt_imm16	bool	0x0	Enable usage of parameter trace_special_hlt_imm16
cpu0.l2cache-hit_latency	int	0x0	L2 Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
cpu0.l2cache-maintenance_latency	int	0x0	L2 Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
cpu0.l2cache-miss_latency	int	0x0	L2 Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
cpu0.l2cache-read_access_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
cpu0.l2cache-read_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per byte accessed.l2cache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
cpu0.l2cache-size	int	0x80000	L2 Cache size in bytes.
cpu0.l2cache-snoop_data_transfer_latency	int	0x0	L2 Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
cpu0.l2cache-snoop_issue_latency	int	0x0	L2 Cache timing annotation latency for snoop accesses issued by this cache in total ticks. This is only used when dcache-state_modelled=true.
cpu0.l2cache-write_access_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-write_latency is set. This is only used when dcache-state_modelled=true.
cpu0.l2cache-write_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per byte accessed. l2cache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-121 Parameters for ARM_Cortex-A55 (continued)

Name	Type	Default value	Description
cpu0.max_code_cache_mb	int	0x100	Maximum size of the simulation code cache (MiB). For platforms with more than 2 cores this limit will be scaled down. (e.g 1/8 for 16 or more cores)
cpu0.min_sync_level	int	0x0	Force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)
cpu0.semihosting-A32_HLT	int	0xf000	A32 HLT number for semihosting calls.
cpu0.semihosting-A64_HLT	int	0xf000	A64 HLT number for semihosting calls.
cpu0.semihosting-ARM_SVC	int	0x123456	A32 SVC number for semihosting calls.
cpu0.semihosting-T32_HLT	int	0x3c	T32 HLT number for semihosting calls.
cpu0.semihosting-Thumb_SVC	int	0xab	T32 SVC number for semihosting calls.
cpu0.semihosting-cmd_line	string	""	Command line available to semihosting calls.
cpu0.semihosting-cwd	string	""	Base directory for semihosting file access.
cpu0.semihosting-enable	bool	0x1	Enable semihosting SVC/HLT traps.
cpu0.semihosting-heap_base	int	0x0	Virtual address of heap base.
cpu0.semihosting-heap_limit	int	0xf000000	Virtual address of top of heap.
cpu0.semihosting-stack_base	int	0x10000000	Virtual address of base of descending stack.
cpu0.semihosting-stack_limit	int	0xf000000	Virtual address of stack limit.
cpu0.trace_special_hlt_imm16	int	0xf000	For this HLT number, IF enable_trace_special_hlt_imm16=true, skip performing usual HLT execution but call MTI trace if registered
cpu0.vfp-enable_at_reset	bool	0x0	Enable VFP in CPACR, CPPWR, NSACR at reset. Warning: Arm recommends going through the implementation's suggested VFP power-up sequence!
cpu0.vfp-present	bool	0x1	Set whether the model has VFP support

Table 3-122 Parameters for Cluster_ARM_Cortex-A55_Cortex-A75

Name	Type	Default value	Description
BROADCASTATOMIC	bool	0x1	Enable broadcasting of atomic operation. The broadcastatomic signal will override this value if used
BROADCASTCACHEMAINT	bool	0x1	Enable broadcasting of cache maintenance operations to downstream caches. The broadcastcachemaint signal will override this value if used.
BROADCASTOUTER	bool	0x1	Enable broadcasting of Outer Shareable transactions. The broadcastouter signal will override this value if used.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-122 Parameters for Cluster_ARM_Cortex-A55_Cortex-A75 (continued)

Name	Type	Default value	Description
BROADCASTPERSIST	bool	0x1	Enable broadcasting of cache clean to the point of persistence operations. The broadcastpersist signal will override this value if used
CLUSTER_ID	int	0x0	Processor cluster ID value
GICDISABLE	bool	0x1	Disable the new style GICv3 CPU interface in each core model. Should be left enabled unless the platform contains a GICv3 distributor.
bus_type	int	0x0	Cosmetic change that changes reset value of L2ACTLR register. 0, ACE. 1, CHI. 2, AXI
dcache-state_modelled	bool	0x0	Set whether D-cache has stateful implementation
default_opmode	int	0x4	Operating mode of DynamIQ coming out of reset. 0: SFONLY ON, 1: 1/4 CACHE ON, 2: 1/2 CACHE ON, 3: 3/4 CACHE ON, 4: FULL CACHE ON
diagnostics	bool	0x0	Enable DynamIQ diagnostic messages
enable_simulation_performance_optimizations	bool	0x1	With this option enabled, the model will run more quickly, but be less accurate to exact CPU behavior. The model will still be functionally accurate for software, but may increase differences seen between hardware behavior and model behavior for certain workloads (it changes the micro-architectural value of stage12_tlb_size parameter to 1024).
has_peripheral_port	bool	0x0	If true, additional AXI peripheral port is configured.
icache-state_modelled	bool	0x0	Set whether I-cache has stateful implementation
l3cache-hit_latency	int	0x0	L3 Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
l3cache-maintenance_latency	int	0x0	L3 Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
l3cache-miss_latency	int	0x0	L3 Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
l3cache-read_access_latency	int	0x0	L3 Cache timing annotation latency for read accesses given in ticks per access (of size l3cache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l3cache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-122 Parameters for Cluster_ARM_Cortex-A55_Cortex-A75 (continued)

Name	Type	Default value	Description
l3cache-read_latency	int	0x0	L3 Cache timing annotation latency for read accesses given in ticks per byte accessed.l3cache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
l3cache-size	int	0x80000	L3 Cache size in bytes.
l3cache-snoop_data_transfer_latency	int	0x0	L3 Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
l3cache-snoop_issue_latency	int	0x0	L3 Cache timing annotation latency for snoop accesses issued by this cache in total ticks. This is only used when dcache-state_modelled=true.
l3cache-write_access_latency	int	0x0	L3 Cache timing annotation latency for write accesses given in ticks per access (of size l3cache-write_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l3cache-write_latency is set. This is only used when dcache-state_modelled=true.
l3cache-write_latency	int	0x0	L3 Cache timing annotation latency for write accesses given in ticks per byte accessed. l3cache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
pchannel_treat_simreset_as_poreset	bool	0x0	Register core as ON state to cluster with simulation reset.
periph_address_end	int	0x0	End address for peripheral port address range exclusive(corresponds to AENDMP input signal).
periph_address_start	int	0x0	Start address for peripheral port address range inclusive(corresponds to ASTARTMP input signal).

Table 3-123 Parameters for Subcluster_ARM_Cortex-A55

Name	Type	Default value	Description
subcluster0.NUM_CORES	int	0x1	Number of cores per cluster
subcluster0.cpi_div	int	0x1	Divider for calculating CPI (Cycles Per Instruction)
subcluster0.cpi_mul	int	0x1	Multiplier for calculating CPI (Cycles Per Instruction)
subcluster0.dcache-hit_latency	int	0x0	L1 D-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-123 Parameters for Subcluster_ARM_Cortex-A55 (continued)

Name	Type	Default value	Description
subcluster0.dcache-maintenance_latency	int	0x0	L1 D-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
subcluster0.dcache-miss_latency	int	0x0	L1 D-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
subcluster0.dcache-prefetch_enabled	bool	0x0	Enable simulation of data cache prefetching. This is only used when dcache-state_modelled=true
subcluster0.dcache-read_access_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per access (of size dcache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
subcluster0.dcache-read_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per byte accessed.dcache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
subcluster0.dcache-size	int	0x8000	L1 D-Cache size in bytes.
subcluster0.dcache-snoop_data_transfer_latency	int	0x0	L1 D-Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
subcluster0.dcache-write_access_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per access (of size dcache-write_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-write_latency is set. This is only used when dcache-state_modelled=true.
subcluster0.dcache-write_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per byte accessed. dcache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
subcluster0.has_dot_product	int	0x2	Possible values of this parameter are: - 1, feature not implemented. - 2, feature is implemented.
subcluster0.icache-hit_latency	int	0x0	L1 I-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when icache-state_modelled=true.
subcluster0.icache-maintenance_latency	int	0x0	L1 I-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when icache-state_modelled=true.
subcluster0.icache-miss_latency	int	0x0	L1 I-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when icache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-123 Parameters for Subcluster_ARM_Cortex-A55 (continued)

Name	Type	Default value	Description
subcluster0.icache-prefetch_enabled	bool	0x0	Enable simulation of instruction cache prefetching. This is only used when icache-state_modelled=true.
subcluster0.icache-read_access_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per access (of size icache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if icache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when icache-state_modelled=true.
subcluster0.icache-read_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per byte accessed.icache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when icache-state_modelled=true.
subcluster0.icache-size	int	0x8000	L1 I-Cache size in bytes.
subcluster0.ptw_latency	int	0x0	Page table walker latency for TA (Timing Annotation), expressed in simulation ticks
subcluster0.tlb_latency	int	0x0	TLB latency for TA (Timing Annotation), expressed in simulation ticks
subcluster0.walk_cache_latency	int	0x0	Walk cache latency for TA (Timing Annotation), expressed in simulation ticks

Table 3-124 Parameters for Subcluster_ARM_Cortex-A75

Name	Type	Default value	Description
subcluster1.CCSIDR-L1D_override	int	0x0	If nonzero, override the value presented in CCSIDR for L1D (this is cosmetic and does not affect cache behaviour).
subcluster1.CCSIDR-L1I_override	int	0x0	If nonzero, override the value presented in CCSIDR for L1I (this is cosmetic and does not affect cache behaviour).
subcluster1.CCSIDR-L2_override	int	0x0	If nonzero, override the value presented in CCSIDR for L2 (this is cosmetic and does not affect cache behaviour).
subcluster1.NUM_CORES	int	0x1	Number of cores per cluster
subcluster1.cpi_div	int	0x1	Divider for calculating CPI (Cycles Per Instruction)
subcluster1.cpi_mul	int	0x1	Multiplier for calculating CPI (Cycles Per Instruction)
subcluster1.dcache-hit_latency	int	0x0	L1 D-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
subcluster1.dcache-maintenance_latency	int	0x0	L1 D-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-124 Parameters for Subcluster_ARM_Cortex-A75 (continued)

Name	Type	Default value	Description
subcluster1.dcache-miss_latency	int	0x0	L1 D-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
subcluster1.dcache-prefetch_enabled	bool	0x0	Enable simulation of data cache prefetching. This is only used when dcache-state_modelled=true
subcluster1.dcache-read_access_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per access (of size dcache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
subcluster1.dcache-read_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per byte accessed.dcache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
subcluster1.dcache-snoop_data_transfer_latency	int	0x0	L1 D-Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
subcluster1.dcache-write_access_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per access (of size dcache-write_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-write_latency is set. This is only used when dcache-state_modelled=true.
subcluster1.dcache-write_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per byte accessed. dcache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
subcluster1.icache-hit_latency	int	0x0	L1 I-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when icache-state_modelled=true.
subcluster1.icache-maintenance_latency	int	0x0	L1 I-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when icache-state_modelled=true.
subcluster1.icache-miss_latency	int	0x0	L1 I-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when icache-state_modelled=true.
subcluster1.icache-prefetch_enabled	bool	0x0	Enable simulation of instruction cache prefetching. This is only used when icache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-124 Parameters for Subcluster_ARM_Cortex-A75 (continued)

Name	Type	Default value	Description
`subcluster1.icache-read_access_latency`	`int`	`0x0`	L1 I-Cache timing annotation latency for read accesses given in ticks per access (of size icache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if icache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when icache-state_modelled=true.
`subcluster1.icache-read_latency`	`int`	`0x0`	L1 I-Cache timing annotation latency for read accesses given in ticks per byte accessed.icache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when icache-state_modelled=true.
`subcluster1.ptw_latency`	`int`	`0x0`	Page table walker latency for TA (Timing Annotation), expressed in simulation ticks
`subcluster1.tlb_latency`	`int`	`0x0`	TLB latency for TA (Timing Annotation), expressed in simulation ticks
`subcluster1.walk_cache_latency`	`int`	`0x0`	Walk cache latency for TA (Timing Annotation), expressed in simulation ticks

Related information

Arm DynamIQ Shared Unit Technical Reference Manual

3.4.18 ARMCortexA55CT_CortexA76CT

ARMCortexA55CT_CortexA76CT CPU component. This model is written in C++.

ARMCortexA55CT_CortexA76CT contains the following CADI targets:

- ARM_Cortex-A55
- Cluster_ARM_Cortex-A55_Cortex-A76
- Subcluster_ARM_Cortex-A55
- Subcluster_ARM_Cortex-A76

ARMCortexA55CT_CortexA76CT contains the following MTI components:

- *4.12 ARM_Cortex-A55 - trace* on page 4-1190
- *4.21 ARM_Cortex-A76 - trace* on page 4-1517
- *4.48 ARMv8Cluster - trace* on page 4-2401
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402
- *4.68 DSU - trace* on page 4-2463
- *4.73 GICv3CPUInterfaceDecoder - trace* on page 4-2489
- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.108 PVCache - trace* on page 4-2761
- *4.114 TLB - trace* on page 4-2834

About ARMCortexA55CT_CortexA76CT

For information about the cores in this model, see:

- *3.4.16 ARMCortexA55CT* on page 3-342.
- *3.4.27 ARMCortexA76CT* on page 3-454.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-125 Ports

Name	Protocol	Type	Description
AENDMP	*Value_64* on page 2-85	Slave	DynamIQ port to obtain end address of valid peripheral address range (exclusive).
ASTARTMP	*Value_64* on page 2-85	Slave	DynamIQ port to obtain start address of valid peripheral address range (inclusive).
CNTHPIRQ[8]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTHVIRQ[8]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTPNSIRQ[8]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTPSIRQ[8]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTVIRQ[8]	*Signal* on page 2-85	Master	Timer signals to SOC.
aa64naa32[8]	*Signal* on page 2-85	Slave	Register width after reset.
acp_s	PVBus	Slave	AXI ACP slave port
broadcastatomic	*Signal* on page 2-85	Slave	CHI defined pins.
broadcastcachemaint	*Signal* on page 2-85	Slave	ACE defined pins.
broadcastouter	*Signal* on page 2-85	Slave	ACE defined pins.
broadcastpersist	*Signal* on page 2-85	Slave	CHI defined pins.
cfgend[8]	*Signal* on page 2-85	Slave	This signal if for EE bit initialisation.
cfgsdisable	*Signal* on page 2-85	Slave	This signal disables write access to some secure Interrupt Controller registers.
cfgte[8]	*Signal* on page 2-85	Slave	This signal provides default exception handling state.
clk_in	ClockSignal	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the cluster level components run e.g cluster level timers, caches and pmu.
clusterid	*Value* on page 2-85	Slave	The port reads the value in CPU ID register field, bits[11:8] of the MPIDR.
clusterpmuirq	*Signal* on page 2-85	Master	DynamIQ pmu irq
cntvalueb	*CounterInterface* on page 2-84	Slave	Interface to SoC level counter module.
commirq[8]	*Signal* on page 2-85	Master	Interrupt signal from debug communications channel.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-125 Ports (continued)

Name	Protocol	Type	Description
core_clk_in[8]	ClockSignal	Slave	The clock signal connected to the core_clk_in port is used to determine the rate at which each core executes instructions.
cp15sdisable[8]	*Signal* on page 2-85	Slave	This signal disables write access to some system control processor registers.
cpuporeset[8]	*Signal* on page 2-85	Slave	CPU power on reset. Initializes all the processor logic, including debug logic.
cryptodisable[8]	*Signal* on page 2-85	Slave	Disable cryptography extensions after reset.
cti[8]	*v8EmbeddedCrossTrigger_controlprotocol* on page 2-84	Master	Cross trigger matrix port.
ctidbgirq[8]	*Signal* on page 2-85	Master	Cross Trigger Interface (CTI) interrupt trigger output.
dbgen[8]	*Signal* on page 2-85	Slave	External debug interface.
dbgnopwrdwn[8]	*Signal* on page 2-85	Master	No power-down request.
dbgpwrupreq[8]	*Signal* on page 2-85	Master	Debug power up request.
dev_debug_s	PVBus	Slave	External debug interface.
event	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE.
fiq[8]	*Signal* on page 2-85	Slave	This signal drives the CPUs fast-interrupt handling.
gicv3_redistributor_s[8]	*GICv3Comms* on page 2-84	Slave	GICv3 AXI-stream port.
irq[8]	*Signal* on page 2-85	Slave	This signal drives the CPUs interrupt handling.
memorymapped_debug_s	PVBus	Slave	External debug interface.
niden[8]	*Signal* on page 2-85	Slave	External debug interface.
pchannel_cluster	*PChannel* on page 2-80	Slave	PChannel for cluster.
pchannel_core[8]	*PChannel* on page 2-80	Slave	PChannels for cores
pmuirq[8]	*Signal* on page 2-85	Master	Interrupt signal from performance monitoring unit.
presetdbg	*Signal* on page 2-85	Slave	Initialize the shared debug APB, Cross Trigger Interface (CTI), and Cross Trigger Matrix (CTM) logic.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-125 Ports (continued)

Name	Protocol	Type	Description
pvbus_m0	PVBus	Master	The core will generate bus requests on this port.
pvbus_periph_m	PVBus	Master	The core can generate peripheral bus request on this port.
rei[8]	*Signal* on page 2-85	Slave	Per core RAM Error Interrupt.
reset[8]	*Signal* on page 2-85	Slave	Raising this signal will put the core into reset mode.
rvbaraddr[8]	*Value_64* on page 2-85	Slave	Reset vector base address.
sci_m	SystemCoherencyInterface	Master	System coherency interface port, which is used to take the whole cluster into/out-of coherency domain
sei[8]	*Signal* on page 2-85	Slave	Per core System Error physical pins.
spiden[8]	*Signal* on page 2-85	Slave	External debug interface.
spniden[8]	*Signal* on page 2-85	Slave	External debug interface.
sporeset	*Signal* on page 2-85	Slave	A single cluster-wide power on reset signal for all resettable registers in DynamIQ.
ticks[8]	*InstructionCount* on page 2-84	Master	This port should be connected to one of the two ticks ports on a 'visualisation' component, in order to display a running instruction count.
vcpumntirq[8]	*Signal* on page 2-85	Master	Interrupt signal for virtual CPU maintenance IRQ.
vfiq[8]	*Signal* on page 2-85	Slave	Virtualised FIQ.
vinithi[8]	*Signal* on page 2-85	Slave	This signal controls of the location of the exception vectors at reset.
virq[8]	*Signal* on page 2-85	Slave	Virtualised IRQ.
virtio_s	PVBus	Slave	The virtio coherent port, hooks directly into the L3 system and becomes coherent (assuming attributes are set correctly).
vsei[8]	*Signal* on page 2-85	Slave	Per core virtual System Error physical pins.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-126 Parameters for ARM_Cortex-A55

Name	Type	Default value	Description
cpu0.AA64nAA32	bool	0x1	Register width configuration at reset. 0, AArch32. 1, AArch64.
cpu0.CFGEND	bool	0x0	Endianness configuration at reset. 0, little endian. 1, big endian.
cpu0.CFGTE	bool	0x0	Instruction set state when resetting into AArch32. 0, A32. 1, T32.
cpu0.CP15SDISABLE	bool	0x0	Initialize to disable access to some CP15 registers
cpu0.CRYPTODISABLE	bool	0x0	Disable cryptographic features.
cpu0.RVBARADDR	int	0x0	Value of RVBAR_ELx register.
cpu0.VINITHI	bool	0x0	Reset value of SCTLR.V.
cpu0.enable_trace_special_hlt_imm16	bool	0x0	Enable usage of parameter trace_special_hlt_imm16
cpu0.l2cache-hit_latency	int	0x0	L2 Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
cpu0.l2cache-maintenance_latency	int	0x0	L2 Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
cpu0.l2cache-miss_latency	int	0x0	L2 Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
cpu0.l2cache-read_access_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
cpu0.l2cache-read_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per byte accessed.l2cache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
cpu0.l2cache-size	int	0x80000	L2 Cache size in bytes.
cpu0.l2cache-snoop_data_transfer_latency	int	0x0	L2 Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
cpu0.l2cache-snoop_issue_latency	int	0x0	L2 Cache timing annotation latency for snoop accesses issued by this cache in total ticks. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-126 Parameters for ARM_Cortex-A55 (continued)

Name	Type	Default value	Description
cpu0.l2cache-write_access_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-write_latency is set. This is only used when dcache-state_modelled=true.
cpu0.l2cache-write_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per byte accessed. l2cache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
cpu0.max_code_cache_mb	int	0x100	Maximum size of the simulation code cache (MiB). For platforms with more than 2 cores this limit will be scaled down. (e.g 1/8 for 16 or more cores)
cpu0.min_sync_level	int	0x0	Force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)
cpu0.semihosting-A32_HLT	int	0xf000	A32 HLT number for semihosting calls.
cpu0.semihosting-A64_HLT	int	0xf000	A64 HLT number for semihosting calls.
cpu0.semihosting-ARM_SVC	int	0x123456	A32 SVC number for semihosting calls.
cpu0.semihosting-T32_HLT	int	0x3c	T32 HLT number for semihosting calls.
cpu0.semihosting-Thumb_SVC	int	0xab	T32 SVC number for semihosting calls.
cpu0.semihosting-cmd_line	string	""	Command line available to semihosting calls.
cpu0.semihosting-cwd	string	""	Base directory for semihosting file access.
cpu0.semihosting-enable	bool	0x1	Enable semihosting SVC/HLT traps.
cpu0.semihosting-heap_base	int	0x0	Virtual address of heap base.
cpu0.semihosting-heap_limit	int	0xf000000	Virtual address of top of heap.
cpu0.semihosting-stack_base	int	0x10000000	Virtual address of base of descending stack.
cpu0.semihosting-stack_limit	int	0xf000000	Virtual address of stack limit.
cpu0.trace_special_hlt_imm16	int	0xf000	For this HLT number, IF enable_trace_special_hlt_imm16=true, skip performing usual HLT execution but call MTI trace if registered
cpu0.vfp-enable_at_reset	bool	0x0	Enable VFP in CPACR, CPPWR, NSACR at reset. Warning: Arm recommends going through the implementation's suggested VFP power-up sequence!
cpu0.vfp-present	bool	0x1	Set whether the model has VFP support

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-127 Parameters for Cluster_ARM_Cortex-A55_Cortex-A76

Name	Type	Default value	Description
BROADCASTATOMIC	bool	0x1	Enable broadcasting of atomic operation. The broadcastatomic signal will override this value if used
BROADCASTCACHEMAINT	bool	0x1	Enable broadcasting of cache maintenance operations to downstream caches. The broadcastcachemaint signal will override this value if used.
BROADCASTOUTER	bool	0x1	Enable broadcasting of Outer Shareable transactions. The broadcastouter signal will override this value if used.
BROADCASTPERSIST	bool	0x1	Enable broadcasting of cache clean to the point of persistence operations. The broadcastpersist signal will override this value if used
CLUSTER_ID	int	0x0	Processor cluster ID value
GICDISABLE	bool	0x1	Disable the new style GICv3 CPU interface in each core model. Should be left enabled unless the platform contains a GICv3 distributor.
bus_type	int	0x0	Cosmetic change that changes reset value of L2ACTLR register. 0, ACE. 1, CHI. 2, AXI
dcache-state_modelled	bool	0x0	Set whether D-cache has stateful implementation
default_opmode	int	0x4	Operating mode of DynamIQ coming out of reset. 0: SFONLY ON, 1: 1/4 CACHE ON, 2: 1/2 CACHE ON, 3: 3/4 CACHE ON, 4: FULL CACHE ON
diagnostics	bool	0x0	Enable DynamIQ diagnostic messages
enable_simulation_performance_optimizations	bool	0x1	With this option enabled, the model will run more quickly, but be less accurate to exact CPU behavior. The model will still be functionally accurate for software, but may increase differences seen between hardware behavior and model behavior for certain workloads (it changes the micro-architectural value of stage12_tlb_size parameter to 1024).
has_peripheral_port	bool	0x0	If true, additional AXI peripheral port is configured.
icache-state_modelled	bool	0x0	Set whether I-cache has stateful implementation
l3cache-hit_latency	int	0x0	L3 Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
l3cache-maintenance_latency	int	0x0	L3 Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
l3cache-miss_latency	int	0x0	L3 Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-127 Parameters for Cluster_ARM_Cortex-A55_Cortex-A76 (continued)

Name	Type	Default value	Description
l3cache-read_access_latency	int	0x0	L3 Cache timing annotation latency for read accesses given in ticks per access (of size l3cache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l3cache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
l3cache-read_latency	int	0x0	L3 Cache timing annotation latency for read accesses given in ticks per byte accessed.l3cache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
l3cache-size	int	0x80000	L3 Cache size in bytes.
l3cache-snoop_data_transfer_latency	int	0x0	L3 Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
l3cache-snoop_issue_latency	int	0x0	L3 Cache timing annotation latency for snoop accesses issued by this cache in total ticks. This is only used when dcache-state_modelled=true.
l3cache-write_access_latency	int	0x0	L3 Cache timing annotation latency for write accesses given in ticks per access (of size l3cache-write_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l3cache-write_latency is set. This is only used when dcache-state_modelled=true.
l3cache-write_latency	int	0x0	L3 Cache timing annotation latency for write accesses given in ticks per byte accessed. l3cache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
pchannel_treat_simreset_as_poreset	bool	0x0	Register core as ON state to cluster with simulation reset.
periph_address_end	int	0x0	End address for peripheral port address range exclusive(corresponds to AENDMP input signal).
periph_address_start	int	0x0	Start address for peripheral port address range inclusive(corresponds to ASTARTMP input signal).

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-128 Parameters for Subcluster_ARM_Cortex-A55

Name	Type	Default value	Description
subcluster0.NUM_CORES	int	0x1	Number of cores per cluster
subcluster0.cpi_div	int	0x1	Divider for calculating CPI (Cycles Per Instruction)
subcluster0.cpi_mul	int	0x1	Multiplier for calculating CPI (Cycles Per Instruction)
subcluster0.dcache-hit_latency	int	0x0	L1 D-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
subcluster0.dcache-maintenance_latency	int	0x0	L1 D-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
subcluster0.dcache-miss_latency	int	0x0	L1 D-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
subcluster0.dcache-prefetch_enabled	bool	0x0	Enable simulation of data cache prefetching. This is only used when dcache-state_modelled=true
subcluster0.dcache-read_access_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per access (of size dcache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
subcluster0.dcache-read_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per byte accessed.dcache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
subcluster0.dcache-size	int	0x8000	L1 D-Cache size in bytes.
subcluster0.dcache-snoop_data_transfer_latency	int	0x0	L1 D-Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
subcluster0.dcache-write_access_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per access (of size dcache-write_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-write_latency is set. This is only used when dcache-state_modelled=true.
subcluster0.dcache-write_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per byte accessed. dcache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
subcluster0.has_dot_product	int	0x2	Possible values of this parameter are: - 1, feature not implemented. - 2, feature is implemented.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-128 Parameters for Subcluster_ARM_Cortex-A55 (continued)

Name	Type	Default value	Description
subcluster0.icache-hit_latency	int	0x0	L1 I-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when icache-state_modelled=true.
subcluster0.icache-maintenance_latency	int	0x0	L1 I-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when icache-state_modelled=true.
subcluster0.icache-miss_latency	int	0x0	L1 I-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when icache-state_modelled=true.
subcluster0.icache-prefetch_enabled	bool	0x0	Enable simulation of instruction cache prefetching. This is only used when icache-state_modelled=true.
subcluster0.icache-read_access_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per access (of size icache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if icache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when icache-state_modelled=true.
subcluster0.icache-read_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per byte accessed.icache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when icache-state_modelled=true.
subcluster0.icache-size	int	0x8000	L1 I-Cache size in bytes.
subcluster0.ptw_latency	int	0x0	Page table walker latency for TA (Timing Annotation), expressed in simulation ticks
subcluster0.tlb_latency	int	0x0	TLB latency for TA (Timing Annotation), expressed in simulation ticks
subcluster0.walk_cache_latency	int	0x0	Walk cache latency for TA (Timing Annotation), expressed in simulation ticks

Table 3-129 Parameters for Subcluster_ARM_Cortex-A76

Name	Type	Default value	Description
subcluster1.NUM_CORES	int	0x1	Number of cores per cluster
subcluster1.cpi_div	int	0x1	Divider for calculating CPI (Cycles Per Instruction)
subcluster1.cpi_mul	int	0x1	Multiplier for calculating CPI (Cycles Per Instruction)
subcluster1.dcache-hit_latency	int	0x0	L1 D-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-129 Parameters for Subcluster_ARM_Cortex-A76 (continued)

Name	Type	Default value	Description
subcluster1.dcache-maintenance_latency	int	0x0	L1 D-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
subcluster1.dcache-miss_latency	int	0x0	L1 D-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
subcluster1.dcache-prefetch_enabled	bool	0x0	Enable simulation of data cache prefetching. This is only used when dcache-state_modelled=true
subcluster1.dcache-read_access_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per access (of size dcache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
subcluster1.dcache-read_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per byte accessed.dcache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
subcluster1.dcache-snoop_data_transfer_latency	int	0x0	L1 D-Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
subcluster1.dcache-write_access_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per access (of size dcache-write_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-write_latency is set. This is only used when dcache-state_modelled=true.
subcluster1.dcache-write_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per byte accessed. dcache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
subcluster1.ext_abort_device_read_is_sync	bool	0x0	Synchronous reporting of device-nGnRE read external aborts
subcluster1.ext_abort_device_write_is_sync	bool	0x0	Synchronous reporting of device-nGnRE write external aborts
subcluster1.ext_abort_so_read_is_sync	bool	0x0	Synchronous reporting of device-nGnRnE read external aborts

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-129 Parameters for Subcluster_ARM_Cortex-A76 (continued)

Name	Type	Default value	Description
subcluster1.ext_abort_so_write_is_sync	bool	0x0	Synchronous reporting of device-nGnRnE write external aborts
subcluster1.has_statistical_profiling	bool	0x1	Whether Statistical Based Profiling is implemented
subcluster1.icache-hit_latency	int	0x0	L1 I-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when icache-state_modelled=true.
subcluster1.icache-maintenance_latency	int	0x0	L1 I-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when icache-state_modelled=true.
subcluster1.icache-miss_latency	int	0x0	L1 I-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when icache-state_modelled=true.
subcluster1.icache-prefetch_enabled	bool	0x0	Enable simulation of instruction cache prefetching. This is only used when icache-state_modelled=true.
subcluster1.icache-read_access_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per access (of size icache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if icache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when icache-state_modelled=true.
subcluster1.icache-read_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per byte accessed.icache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when icache-state_modelled=true.
subcluster1.ptw_latency	int	0x0	Page table walker latency for TA (Timing Annotation), expressed in simulation ticks
subcluster1.tlb_latency	int	0x0	TLB latency for TA (Timing Annotation), expressed in simulation ticks
subcluster1.treat-dcache-cmos-to-pou-as-nop	bool	0x0	Whether dcache invalidation to the point of unification is required for instruction to data coherence. true - Invalidate operations not required
subcluster1.walk_cache_latency	int	0x0	Walk cache latency for TA (Timing Annotation), expressed in simulation ticks

3.4.19 ARMCortexA57x1CT

ARMCortexA57x1CT CPU component. This model is written in C++ and models version r0p0 of the RTL.

ARMCortexA57x1CT contains the following CADI targets:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

- ARM_Cortex-A57
- Cluster_ARM_Cortex-A57
- PVCache
- TlbCadi

ARMCortexA57x1CT contains the following MTI components:

- *4.13 ARM_Cortex-A57 - trace* on page 4-1228
- *4.48 ARMv8Cluster - trace* on page 4-2401
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402
- *4.68 DSU - trace* on page 4-2463
- *4.73 GICv3CPUInterfaceDecoder - trace* on page 4-2489
- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.108 PVCache - trace* on page 4-2761
- *4.114 TLB - trace* on page 4-2834

About ARMCortexA57x1CT

- The following components also exist:
  - ARMCortexA57x2CT.
  - ARMCortexA57x3CT.
  - ARMCortexA57x4CT.

  The per-core parameters are preceded by `cpun.`, where *n* identifies the core (0-3).
- The cache latency parameters are only effective when you enable cache-state modeling.
- Timing annotation for transactions downstream of the cache and TLB models propagates through the models.
- This model supports the Armv8-A Cryptography Extensions, which requires the Crypto plug-in to be loaded. See *Crypto plug-in* on page 5-2909 for details.
- The system designer decides whether a debug APB ties the external debug view with other system views. In the model, use a `PVBusDecoder` to direct traffic to the correct port, `dev_debug_s` or `memorymapped_debug_s`.

Differences between the model and the RTL

This component has the following differences from the corresponding revision of the RTL implementation:

- The value of the AArch64 PMCEID0_EL0 register, and the AArch32 alias of this register, differs in the model from the TRM value. The model value reflects the model counters.
- The mechanisms for setting the affinity fields of the MPIDR. The RTL has two ports:
  - CLUSTERIDAFF1[7:0].
  - CLUSTERIDAFF2[7:0].

  AFF1 sets the value of MPIDR bits[15:8] and AFF2 sets the value of MPIDR bits[23:16]. In contrast, the model has a single `CLUSTER_ID` port. This difference allows the setting of bits[23:8] of the MPIDR using bits[15:0] of the `CLUSTER_ID` value.
- The memory mapped debug registers have a view for cores and a view for external debug agents. In the model, these views require two PVBus ports. In hardware, the system designer decides how the implementation differentiates the views.
- In the model, a single `peer` event port combines the functionality of the `eventi` and `evento` signals in the RTL.
- The Generic Timers are *Programmer's View* (PV) level abstractions: a model-specific protocol connects the `cntvalueb` port to the MemoryMappedCounterModule.
- The GIC CPU Interface is a PV level abstraction: a model-specific protocol connects the GIC CPU Interface to the GIC Distributor.
- The CoreSight *Cross Trigger Interface* (CTI) is a PV level abstraction: the interface is a model-specific one.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

- The model has no mechanism to read the internal memory that the Cache and TLB structures use, through the implementation defined region of the system coprocessor interface. This memory includes the RAM Index Register, IL1DATA Registers, DL1DATA Registers, and associated functionality.
- The model does not implement:
  - ETM registers.
  - The PMUEVENT bus.
  - The WARMRESETREQ signal. However, the warm reset code sequence (see the section *Code sequence to request a Warm reset as a result of RMR_ELx.RR* in the *Arm®v8-A Architecture Reference Manual*) makes the model simulate a warm reset of the core.
  - The PMUSNAPSHOTREQ and PMUSNAPSHOTACK signals.
  - The EXTERRIRQ and INTERRIRQ signals.
  - Processor dynamic-retention signals.
  - The SYSBARDISABLE signal.
  - The DBGPWRDUP, DBGPWRUPREQ, DBGNOPWRDWN, and DBGRSTREQ debug power management signals.
  - ECC and parity schemes are hardware-specific so are not supported.

Table 3-130 Ports

Name	Protocol	Type	Description
CNTHPIRQ[4]	*Signal* on page 2-85	Master	The per-EL counter signal.
CNTPNSIRQ[4]	*Signal* on page 2-85	Master	The per-EL counter signal.
CNTPSIRQ[4]	*Signal* on page 2-85	Master	The per-EL counter signal.
CNTVIRQ[4]	*Signal* on page 2-85	Master	The per-EL counter signal.
aa64naa32[4]	*Signal* on page 2-85	Slave	Register width after reset.
acp_s	PVBus	Slave	AXI ACP slave port.
broadcastcachemaint	*Signal* on page 2-85	Slave	Enable broadcasting of cache maintenance operations to downstream caches.
broadcastinner	*Signal* on page 2-85	Slave	Enable broadcasting of Inner Shareable transactions.
broadcastouter	*Signal* on page 2-85	Slave	Enable broadcasting of Outer Shareable transactions.
cfgend[4]	*Signal* on page 2-85	Slave	This signal if for EE bit initialisation.
cfgsdisable	*Signal* on page 2-85	Slave	This signal disables write access to some secure Interrupt Controller registers.
cfgte[4]	*Signal* on page 2-85	Slave	This signal provides default exception handling state.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-130 Ports (continued)

Name	Protocol	Type	Description
clk_in	ClockSignal	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the core executes instructions.
clrexmonack	*Signal* on page 2-85	Master	Acknowledge handshake signal for the clrexmonreq signal
clrexmonreq	*Signal* on page 2-85	Slave	Signals the clearing of an external global exclusive monitor
clusterid	*Value* on page 2-85	Slave	The port reads the value in CPU ID register field, bits[11:8] of the MPIDR.
cntvalueb	*CounterInterface* on page 2-84	Slave	Interface to SoC level counter module.
commirq[4]	*Signal* on page 2-85	Master	Interrupt signal from debug communications channel.
commrx[4]	*Signal* on page 2-85	Master	Receive portion of Data Transfer Register full.
commtx[4]	*Signal* on page 2-85	Master	Transmit portion of Data Transfer Register empty.
cp15sdisable[4]	*Signal* on page 2-85	Slave	This signal disables write access to some system control processor registers.
cpuporeset[4]	*Signal* on page 2-85	Slave	Power on reset. Initializes all the processor logic, including debug logic.
cryptodisable[4]	*Signal* on page 2-85	Slave	Disable cryptography extensions after reset.
cti[4]	*v8EmbeddedCrossTrigger_controlprotocol* on page 2-84	Master	Cross trigger matrix port.
ctidbgirq[4]	*Signal* on page 2-85	Master	Cross Trigger Interface (CTI) interrupt trigger output.
dbgack[4]	*Signal* on page 2-85	Master	External debug interface.
dbgen[4]	*Signal* on page 2-85	Slave	External debug interface.
dbgnopwrdwn[4]	*Signal* on page 2-85	Master	These signals relate to core power down.
dbgpwrupreq[4]	*Signal* on page 2-85	Master	These signals relate to core power down.
dev_debug_s	PVBus	Slave	External debug interface.
edbgrq[4]	*Signal* on page 2-85	Slave	External debug interface.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-130 Ports (continued)

Name	Protocol	Type	Description
event	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE.
fiq[4]	*Signal* on page 2-85	Slave	This signal drives the CPUs fast-interrupt handling.
gicv3_redistributor_s[4]	*GICv3Comms* on page 2-84	Slave	GICv3 AXI-stream port.
irq[4]	*Signal* on page 2-85	Slave	This signal drives the CPUs interrupt handling.
l2flushdone	*Signal* on page 2-85	Master	Flush of L2 memory system complete
l2flushreq	*Signal* on page 2-85	Slave	Request flush of L2 memory system.
l2reset	*Signal* on page 2-85	Slave	Reset the shared L2 memory system controller.
memorymapped_debug_s	PVBus	Slave	External debug interface.
niden[4]	*Signal* on page 2-85	Slave	External debug interface.
periphbase	*Value_64* on page 2-85	Slave	This port sets the base address of private peripheral region.
pmuirq[4]	*Signal* on page 2-85	Master	Interrupt signal from performance monitoring unit.
presetdbg	*Signal* on page 2-85	Slave	Initialize the shared debug APB, Cross Trigger Interface (CTI), and Cross Trigger Matrix (CTM) logic.
pvbus_m0	PVBus	Master	The core will generate bus requests on this port.
rei[4]	*Signal* on page 2-85	Slave	Individual processor RAM Error Interrupt signal input.
reset[4]	*Signal* on page 2-85	Slave	Raising this signal will put the core into reset mode.
romaddr	*Value_64* on page 2-85	Slave	Debug ROM base address.
romaddrv	*Signal* on page 2-85	Slave	Debug ROM base address valid.
rvbaraddr[4]	*Value_64* on page 2-85	Slave	Reset vector base address.
sei[4]	*Signal* on page 2-85	Slave	Per core System Error physical pins.
smpen[4]	*Signal* on page 2-85	Master	This signals AMP or SMP mode for each core.
spiden[4]	*Signal* on page 2-85	Slave	External debug interface.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-130 Ports (continued)

Name	Protocol	Type	Description
spniden[4]	*Signal* on page 2-85	Slave	External debug interface.
standbywfe[4]	*Signal* on page 2-85	Master	This signal indicates if a core is in WFE state.
standbywfi[4]	*Signal* on page 2-85	Master	This signal indicates if a core is in WFI state.
standbywfil2	*Signal* on page 2-85	Master	Indicate that all the individual processors and the L2 systems are in a WFI state.
ticks[4]	*InstructionCount* on page 2-84	Master	This port should be connected to one of the two ticks ports on a 'visualisation' component, in order to display a running instruction count.
vcpumntirq[4]	*Signal* on page 2-85	Master	Interrupt signal for virtual CPU maintenance IRQ.
vfiq[4]	*Signal* on page 2-85	Slave	Virtual FIQ.
vinithi[4]	*Signal* on page 2-85	Slave	This signal controls of the location of the exception vectors at reset.
virq[4]	*Signal* on page 2-85	Slave	Virtual IRQ.
virtio_s	PVBus	Slave	The virtio coherent port, hooks directly into the L2 system and becomes coherent (assuming attributes are set correctly).
vsei[4]	*Signal* on page 2-85	Slave	Processor Virtual System Error Interrupt request.

Table 3-131 Parameters for ARM_Cortex-A57

Name	Type	Default value	Description
cpu0.AA64nAA32	bool	0x1	Register width configuration at reset. 0, AArch32. 1, AArch64.
cpu0.CFGEND	bool	0x0	Endianness configuration at reset. 0, little endian. 1, big endian.
cpu0.CFGTE	bool	0x0	Instruction set state when resetting into AArch32. 0, A32. 1, T32.
cpu0.CP15SDISABLE	bool	0x0	Initialize to disable access to some CP15 registers
cpu0.CRYPTODISABLE	bool	0x0	Disable cryptographic features.
cpu0.RVBARADDR	int	0x0	Value of RVBAR_ELx register.
cpu0.VINITHI	bool	0x0	Reset value of SCTLR.V.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-131 Parameters for ARM_Cortex-A57 (continued)

Name	Type	Default value	Description
cpu0.enable_trace_special_hlt_imm16	bool	0x0	Enable usage of parameter trace_special_hlt_imm16
cpu0.max_code_cache_mb	int	0x100	Maximum size of the simulation code cache (MiB). For platforms with more than 2 cores this limit will be scaled down. (e.g 1/8 for 16 or more cores)
cpu0.min_sync_level	int	0x0	Force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)
cpu0.semihosting-A32_HLT	int	0xf000	A32 HLT number for semihosting calls.
cpu0.semihosting-A64_HLT	int	0xf000	A64 HLT number for semihosting calls.
cpu0.semihosting-ARM_SVC	int	0x123456	A32 SVC number for semihosting calls.
cpu0.semihosting-T32_HLT	int	0x3c	T32 HLT number for semihosting calls.
cpu0.semihosting-Thumb_SVC	int	0xab	T32 SVC number for semihosting calls.
cpu0.semihosting-cmd_line	string	""	Command line available to semihosting calls.
cpu0.semihosting-cwd	string	""	Base directory for semihosting file access.
cpu0.semihosting-enable	bool	0x1	Enable semihosting SVC/HLT traps.
cpu0.semihosting-heap_base	int	0x0	Virtual address of heap base.
cpu0.semihosting-heap_limit	int	0xf000000	Virtual address of top of heap.
cpu0.semihosting-stack_base	int	0x10000000	Virtual address of base of descending stack.
cpu0.semihosting-stack_limit	int	0xf000000	Virtual address of stack limit.
cpu0.trace_special_hlt_imm16	int	0xf000	For this HLT number, IF enable_trace_special_hlt_imm16=true, skip performing usual HLT execution but call MTI trace if registered
cpu0.vfp-enable_at_reset	bool	0x0	Enable VFP in CPACR, CPPWR, NSACR at reset. Warning: Arm recommends going through the implementation's suggested VFP power-up sequence!

Table 3-132 Parameters for Cluster_ARM_Cortex-A57

Name	Type	Default value	Description
BROADCASTCACHEMAINT	bool	0x1	Enable broadcasting of cache maintenance operations to downstream caches. The broadcastcachemaint signal will override this value if used.
BROADCASTINNER	bool	0x1	Enable broadcasting of Inner Shareable transactions. The broadcastinner signal will override this value if used.
BROADCASTOUTER	bool	0x1	Enable broadcasting of Outer Shareable transactions. The broadcastouter signal will override this value if used.
CLUSTER_ID	int	0x0	Processor cluster ID value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-132 Parameters for Cluster_ARM_Cortex-A57 (continued)

Name	Type	Default value	Description
DBGROMADDR	int	0x0	Initialization value of DBGDRAR register. Bits[47:12] of this register specify the ROM table physical address.
DBGROMADDRV	bool	0x0	If true, set bits[1:0] of the CP15 DBGDRAR to indicate that the address is valid
GICDISABLE	bool	0x1	Disable the new style GICv3 CPU interface in each core model. Should be left enabled unless the platform contains a GICv3 distributor.
PERIPHBASE	int	0x13080000	Base address of peripheral memory space
bus_type	int	0x0	Cosmetic change that changes reset value of L2ACTLR register. 0, ACE. 1, CHI. 2, AXI
cpi_div	int	0x1	Divider for calculating CPI (Cycles Per Instruction)
cpi_mul	int	0x1	Multiplier for calculating CPI (Cycles Per Instruction)
dcache-hit_latency	int	0x0	L1 D-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
dcache-maintenance_latency	int	0x0	L1 D-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
dcache-miss_latency	int	0x0	L1 D-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
dcache-prefetch_enabled	bool	0x0	Enable simulation of data cache prefetching. This is only used when dcache-state_modelled=true
dcache-read_access_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per access (of size dcache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
dcache-read_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per byte accessed.dcache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-132 Parameters for Cluster_ARM_Cortex-A57 (continued)

Name	Type	Default value	Description
dcache-snoop_data_transfer_latency	int	0x0	L1 D-Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
dcache-state_modelled	bool	0x0	Set whether D-cache has stateful implementation
dcache-write_access_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per access (of size dcache-write_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-write_latency is set. This is only used when dcache-state_modelled=true.
dcache-write_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per byte accessed. dcache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
enable_simulation_performance_optimizations	bool	0x1	With this option enabled, the model will run more quickly, but be less accurate to exact CPU behavior. The model will still be functionally accurate for software, but may increase differences seen between hardware behavior and model behavior for certain workloads (it changes the micro-architectural value of stage12_tlb_size parameter to 1024).
icache-hit_latency	int	0x0	L1 I-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when icache-state_modelled=true.
icache-maintenance_latency	int	0x0	L1 I-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when icache-state_modelled=true.
icache-miss_latency	int	0x0	L1 I-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when icache-state_modelled=true.
icache-prefetch_enabled	bool	0x0	Enable simulation of instruction cache prefetching. This is only used when icache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-132 Parameters for Cluster_ARM_Cortex-A57 (continued)

Name	Type	Default value	Description
icache-read_access_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per access (of size icache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if icache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when icache-state_modelled=true.
icache-read_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per byte accessed.icache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when icache-state_modelled=true.
icache-state_modelled	bool	0x0	Set whether I-cache has stateful implementation
l2cache-hit_latency	int	0x0	L2 Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
l2cache-maintenance_latency	int	0x0	L2 Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
l2cache-miss_latency	int	0x0	L2 Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
l2cache-read_access_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
l2cache-read_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per byte accessed.l2cache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
l2cache-size	int	0x80000	L2 Cache size in bytes.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-132 Parameters for Cluster_ARM_Cortex-A57 (continued)

Name	Type	Default value	Description
`l2cache-snoop_data_transfer_latency`	`int`	`0x0`	L2 Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
`l2cache-snoop_issue_latency`	`int`	`0x0`	L2 Cache timing annotation latency for snoop accesses issued by this cache in total ticks. This is only used when dcache-state_modelled=true.
`l2cache-write_access_latency`	`int`	`0x0`	L2 Cache timing annotation latency for write accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-write_latency is set. This is only used when dcache-state_modelled=true.
`l2cache-write_latency`	`int`	`0x0`	L2 Cache timing annotation latency for write accesses given in ticks per byte accessed. l2cache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
`ptw_latency`	`int`	`0x0`	Page table walker latency for TA (Timing Annotation), expressed in simulation ticks
`tlb_latency`	`int`	`0x0`	TLB latency for TA (Timing Annotation), expressed in simulation ticks
`walk_cache_latency`	`int`	`0x0`	Walk cache latency for TA (Timing Annotation), expressed in simulation ticks

3.4.20 ARMCortexA65AECT

ARMCortexA65AECT CPU component. This model is written in C++.

ARMCortexA65AECT contains the following CADI targets:

- ARM_Cortex-A65AE
- Cluster_ARM_Cortex-A65AE
- PVCache
- TlbCadi

ARMCortexA65AECT contains the following MTI components:

- *4.16 ARM_Cortex-A65AE - trace* on page 4-1329
- *4.48 ARMv8Cluster - trace* on page 4-2401
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402
- *4.68 DSU - trace* on page 4-2463
- *4.73 GICv3CPUInterfaceDecoder - trace* on page 4-2489
- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.108 PVCache - trace* on page 4-2761
- *4.114 TLB - trace* on page 4-2834

About ARMCortexA65AECT

The model supports the following features:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

- DynamIQ (FCM) system registers.
- Per-core L2 cache.
- A P-Channel for the cluster and for each core.
- Optional peripheral port.
- L3 cache partition.
- Per-core clock.
- `BROADCASTPERSIST` pin.

The model does not support the following features:

- `BROADCASTCACHEMAINTPOU` pin.
- `COREINSTRRET`, `COREINSTRRUN`, or `nPMBIRQ` signals.
- DynamIQ features that are irrelevant to the programmers' view simulation, for example:
  - Automatic CPU retention mode.
  - Level-3 Cache RAM retention.
- 256-bit wide output transactions.
- ECC and parity schemes are hardware-specific so are not supported.
- Self-test features (MBIST).
- Latency configuration.
- Snoop filtering.
- Debug power management signals `DBGPWRDUP` and `DBGRSTREQ` are not implemented, but `DBGPWRUPREQ` and `DBGNOPWRDWN` are implemented.
- Cache stashing capability.
- Each thread has its own L1Cache and L2Cache instead of a common cache that is shared by all threads of the core.
- Per-thread parameters are not yet supported but signals are implemented.
- Split/Lock

This model has a variable number of cores per cluster, specified using the `NUM_CORES` parameter. The following models also exist, with a fixed number of cores per cluster:

- ARMCortexA65AECTx2CT.
- ARMCortexA65AECTx4CT.
- ARMCortexA65AECTx6CT.
- ARMCortexA65AECTx8CT.

The per-core parameters are preceded by `cpu`*n*`.`, where *n* identifies the core (0-7).

Table 3-133 Ports

Name	Protocol	Type	Description
AENDMP	*Value_64* on page 2-85	Slave	Port to obtain end address of valid peripheral address range (exclusive).
ASTARTMP	*Value_64* on page 2-85	Slave	Port to obtain start address of valid peripheral address range (inclusive).
CNTHPIRQ[16]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTHVIRQ[16]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTPNSIRQ[16]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTPSIRQ[16]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTVIRQ[16]	*Signal* on page 2-85	Master	Timer signals to SOC.
acp_s	PVBus	Slave	AXI ACP slave port
broadcastatomic	*Signal* on page 2-85	Slave	CHI defined pins.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-133 Ports (continued)

Name	Protocol	Type	Description
broadcastcachemaint	*Signal* on page 2-85	Slave	ACE defined pins.
broadcastouter	*Signal* on page 2-85	Slave	ACE defined pins.
broadcastpersist	*Signal* on page 2-85	Slave	CHI defined pins.
cfgend[16]	*Signal* on page 2-85	Slave	This signal if for EE bit initialisation.
cfgte[16]	*Signal* on page 2-85	Slave	This signal provides default exception handling state.
clk_in	ClockSignal	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the core executes instructions.
clusterid	*Value* on page 2-85	Slave	The port reads the value in CPU ID register field, bits[11:8] of the MPIDR.
clusterpmuirq	*Signal* on page 2-85	Master	DynamIQ pmu irq
cntvalueb	*CounterInterface* on page 2-84	Slave	Interface to SoC level counter module.
commirq[16]	*Signal* on page 2-85	Master	Interrupt signal from debug communications channel.
core_clk_in[8]	ClockSignal	Slave	The clock signal connected to the core_clk_in port is used to determine the rate at which each core executes instructions.
cpuporeset[8]	*Signal* on page 2-85	Slave	Power on reset. Initializes all the processor logic, including debug logic.
cryptodisable[16]	*Signal* on page 2-85	Slave	Disable cryptography extensions after reset.
cti[16]	*v8EmbeddedCrossTrigger_controlprotocol* on page 2-84	Master	Cross trigger matrix port.
ctidbgirq[16]	*Signal* on page 2-85	Master	Cross Trigger Interface (CTI) interrupt trigger output.
dbgen	*Signal* on page 2-85	Slave	External debug interface.
dbgnopwrdwn[16]	*Signal* on page 2-85	Master	These signals relate to core power down.
dbgpwrupreq[16]	*Signal* on page 2-85	Master	Debug power up request.
dev_debug_s	PVBus	Slave	External debug interface.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-133 Ports (continued)

Name	Protocol	Type	Description
event	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE.
fiq[16]	*Signal* on page 2-85	Slave	This signal drives the CPUs fast-interrupt handling.
gicv3_redistributor_s[16]	*GICv3Comms* on page 2-84	Slave	GICv3 AXI-stream port per thread.
irq[16]	*Signal* on page 2-85	Slave	This signal drives the CPUs interrupt handling.
memorymapped_debug_s	PVBus	Slave	External debug interface.
niden	*Signal* on page 2-85	Slave	External debug interface.
pchannel_cluster	*PChannel* on page 2-80	Slave	PChannel for cluster.
pchannel_core[8]	*PChannel* on page 2-80	Slave	PChannels for cores
pmuirq[16]	*Signal* on page 2-85	Master	Interrupt signal from performance monitoring unit.
presetdbg	*Signal* on page 2-85	Slave	Initialize the shared debug APB, Cross Trigger Interface (CTI), and Cross Trigger Matrix (CTM) logic.
pvbus_m0	PVBus	Master	The core will generate bus requests on this port.
pvbus_periph_m	PVBus	Master	The core can generate peripheral bus request on this port.
rei[16]	*Signal* on page 2-85	Slave	Per core RAM Error Interrupt.
reset[8]	*Signal* on page 2-85	Slave	Raising this signal will put the core into reset mode.
rvbaraddr[16]	*Value_64* on page 2-85	Slave	Reset vector base address.
sci_m	SystemCoherencyInterface	Master	System coherency interface port, which is used to take the whole cluster into/out-of coherency domain
sei[16]	*Signal* on page 2-85	Slave	Per core virtual System Error physical pins.
spiden	*Signal* on page 2-85	Slave	External debug interface.
spniden	*Signal* on page 2-85	Slave	External debug interface.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-133 Ports (continued)

Name	Protocol	Type	Description
sporeset	*Signal* on page 2-85	Slave	A single cluster-wide power on reset signal for all resettable registers in DynamIQ.
ticks[16]	*InstructionCount* on page 2-84	Master	This port should be connected to one of the two ticks ports on a 'visualisation' component, in order to display a running instruction count.
vcpumntirq[16]	*Signal* on page 2-85	Master	Interrupt signal for virtual CPU maintenance IRQ.
vfiq[16]	*Signal* on page 2-85	Slave	Virtualised FIQ.
vinithi[16]	*Signal* on page 2-85	Slave	This signal controls of the location of the exception vectors at reset.
virq[16]	*Signal* on page 2-85	Slave	Virtualised IRQ.
virtio_s	PVBus	Slave	The virtio coherent port, hooks directly into the L2 system and becomes coherent (assuming attributes are set correctly).
vsei[16]	*Signal* on page 2-85	Slave	Per core virtual System Error physical pins.

Table 3-134 Parameters for Cluster_ARM_Cortex-A65AE

Name	Type	Default value	Description
BROADCASTATOMIC	bool	0x1	Enable broadcasting of atomic operation. The broadcastatomic signal will override this value if used
BROADCASTCACHEMAINT	bool	0x1	Enable broadcasting of cache maintenance operations to downstream caches. The broadcastcachemaint signal will override this value if used.
BROADCASTOUTER	bool	0x1	Enable broadcasting of Outer Shareable transactions. The broadcastouter signal will override this value if used.
BROADCASTPERSIST	bool	0x1	Enable broadcasting of cache clean to the point of persistence operations. The broadcastpersist signal will override this value if used
CLUSTER_ID	int	0x0	Processor cluster ID value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-134 Parameters for Cluster_ARM_Cortex-A65AE (continued)

Name	Type	Default value	Description
GICDISABLE	bool	0x1	Disable the new style GICv3 CPU interface in each core model. Should be left enabled unless the platform contains a GICv3 distributor.
NUM_CORES	int	0x2	Number of cores per cluster
cluster_patch_level	int	0x0	Cosmetic change to patch number in CLUSTERIDR. Corresponds to the Y in rXpY.
cluster_revision_number	int	0x0	Cosmetic change to revision number in CLUSTERIDR, Corresponds to the X in rXpY.
cpi_div	int	0x1	Divider for calculating CPI (Cycles Per Instruction)
cpi_mul	int	0x1	Multiplier for calculating CPI (Cycles Per Instruction)
dcache-hit_latency	int	0x0	L1 D-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
dcache-maintenance_latency	int	0x0	L1 D-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
dcache-miss_latency	int	0x0	L1 D-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
dcache-prefetch_enabled	bool	0x0	Enable simulation of data cache prefetching. This is only used when dcache-state_modelled=true
dcache-read_access_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per access (of size dcache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
dcache-read_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per byte accessed.dcache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
dcache-size	int	0x10000	L1 D-Cache size in bytes.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-134 Parameters for Cluster_ARM_Cortex-A65AE (continued)

Name	Type	Default value	Description
dcache-snoop_data_transfer_latency	int	0x0	L1 D-Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
dcache-state_modelled	bool	0x0	Set whether D-cache has stateful implementation
dcache-write_access_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per access (of size dcache-write_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-write_latency is set. This is only used when dcache-state_modelled=true.
dcache-write_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per byte accessed. dcache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
default_opmode	int	0x4	Operating mode of DynamIQ coming out of reset. 0: SFONLY ON, 1: 1/4 CACHE ON, 2: 1/2 CACHE ON, 3: 3/4 CACHE ON, 4: FULL CACHE ON
diagnostics	bool	0x0	Enable DynamIQ diagnostic messages
enable_simulation_performance_optimizations	bool	0x1	With this option enabled, the model will run more quickly, but be less accurate to exact CPU behavior. The model will still be functionally accurate for software, but may increase differences seen between hardware behavior and model behavior for certain workloads (it changes the micro-architectural value of stage12_tlb_size parameter to 1024).
has_dot_product	int	0x2	Possible values of this parameter are: - 1, feature not implemented. - 2, feature is implemented.
has_peripheral_port	bool	0x0	If true, additional AXI peripheral port is configured.
icache-hit_latency	int	0x0	L1 I-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when icache-state_modelled=true.
icache-maintenance_latency	int	0x0	L1 I-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when icache-state_modelled=true.
icache-miss_latency	int	0x0	L1 I-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when icache-state_modelled=true.
icache-prefetch_enabled	bool	0x0	Enable simulation of instruction cache prefetching. This is only used when icache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-134 Parameters for Cluster_ARM_Cortex-A65AE (continued)

Name	Type	Default value	Description
icache-read_access_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per access (of size icache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if icache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when icache-state_modelled=true.
icache-read_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per byte accessed.icache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when icache-state_modelled=true.
icache-size	int	0x10000	L1 I-Cache size in bytes.
icache-state_modelled	bool	0x0	Set whether I-cache has stateful implementation
l3cache-hit_latency	int	0x0	L3 Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
l3cache-maintenance_latency	int	0x0	L3 Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
l3cache-miss_latency	int	0x0	L3 Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
l3cache-read_access_latency	int	0x0	L3 Cache timing annotation latency for read accesses given in ticks per access (of size l3cache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l3cache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
l3cache-read_latency	int	0x0	L3 Cache timing annotation latency for read accesses given in ticks per byte accessed.l3cache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-134 Parameters for Cluster_ARM_Cortex-A65AE (continued)

Name	Type	Default value	Description
l3cache-size	int	0x400000	L3 Cache size in bytes.
l3cache-snoop_data_transfer_latency	int	0x0	L3 Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
l3cache-snoop_issue_latency	int	0x0	L3 Cache timing annotation latency for snoop accesses issued by this cache in total ticks. This is only used when dcache-state_modelled=true.
l3cache-write_access_latency	int	0x0	L3 Cache timing annotation latency for write accesses given in ticks per access (of size l3cache-write_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l3cache-write_latency is set. This is only used when dcache-state_modelled=true.
l3cache-write_latency	int	0x0	L3 Cache timing annotation latency for write accesses given in ticks per byte accessed. l3cache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
pchannel_treat_simreset_as_poreset	bool	0x0	Register core as ON state to cluster with simulation reset.
periph_address_end	int	0x0	End address for peripheral port address range exclusive(corresponds to AENDMP input signal).
periph_address_start	int	0x0	Start address for peripheral port address range inclusive(corresponds to ASTARTMP input signal).
ptw_latency	int	0x0	Page table walker latency for TA (Timing Annotation), expressed in simulation ticks
tlb_latency	int	0x0	TLB latency for TA (Timing Annotation), expressed in simulation ticks
walk_cache_latency	int	0x0	Walk cache latency for TA (Timing Annotation), expressed in simulation ticks

Table 3-135 Parameters for ARM_Cortex-A65AE

Name	Type	Default value	Description
cpu0.thread0.CRYPTODISABLE	bool	0x0	Disable cryptographic features.
cpu0.thread0.enable_single_thread_at_reset	bool	0x0	Enable single thread after reset and keep other thread in reset
cpu0.thread0.enable_trace_special_hlt_imm16	bool	0x0	Enable usage of parameter trace_special_hlt_imm16

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-135 Parameters for ARM_Cortex-A65AE (continued)

Name	Type	Default value	Description
cpu0.thread0.l2cache-hit_latency	int	0x0	L2 Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
cpu0.thread0.l2cache-maintenance_latency	int	0x0	L2 Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
cpu0.thread0.l2cache-miss_latency	int	0x0	L2 Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
cpu0.thread0.l2cache-read_access_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
cpu0.thread0.l2cache-read_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per byte accessed.l2cache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
cpu0.thread0.l2cache-size	int	0x40000	L2 Cache size in bytes.
cpu0.thread0.l2cache-snoop_data_transfer_latency	int	0x0	L2 Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
cpu0.thread0.l2cache-snoop_issue_latency	int	0x0	L2 Cache timing annotation latency for snoop accesses issued by this cache in total ticks. This is only used when dcache-state_modelled=true.
cpu0.thread0.l2cache-write_access_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-write_latency is set. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-135 Parameters for ARM_Cortex-A65AE (continued)

Name	Type	Default value	Description
cpu0.thread0.l2cache-write_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per byte accessed. l2cache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
cpu0.thread0.max_code_cache_mb	int	0x100	Maximum size of the simulation code cache (MiB). For platforms with more than 2 cores this limit will be scaled down. (e.g 1/8 for 16 or more cores)
cpu0.thread0.min_sync_level	int	0x0	Force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)
cpu0.thread0.semihosting-A32_HLT	int	0xf000	A32 HLT number for semihosting calls.
cpu0.thread0.semihosting-A64_HLT	int	0xf000	A64 HLT number for semihosting calls.
cpu0.thread0.semihosting-ARM_SVC	int	0x123456	A32 SVC number for semihosting calls.
cpu0.thread0.semihosting-T32_HLT	int	0x3c	T32 HLT number for semihosting calls.
cpu0.thread0.semihosting-Thumb_SVC	int	0xab	T32 SVC number for semihosting calls.
cpu0.thread0.semihosting-cmd_line	string	""	Command line available to semihosting calls.
cpu0.thread0.semihosting-cwd	string	""	Base directory for semihosting file access.
cpu0.thread0.semihosting-enable	bool	0x1	Enable semihosting SVC/HLT traps.
cpu0.thread0.semihosting-heap_base	int	0x0	Virtual address of heap base.
cpu0.thread0.semihosting-heap_limit	int	0xf000000	Virtual address of top of heap.
cpu0.thread0.semihosting-stack_base	int	0x10000000	Virtual address of base of descending stack.
cpu0.thread0.semihosting-stack_limit	int	0xf000000	Virtual address of stack limit.
cpu0.thread0.trace_special_hlt_imm16	int	0xf000	For this HLT number, IF enable_trace_special_hlt_imm16=true, skip performing usual HLT execution but call MTI trace if registered
cpu0.thread0.vfp-enable_at_reset	bool	0x0	Enable VFP in CPACR, CPPWR, NSACR at reset. Warning: Arm recommends going through the implementation's suggested VFP power-up sequence!
cpu0.thread0.vfp-present	bool	0x1	Set whether the model has VFP support

3.4.21 ARMCortexA65AECT_CortexA76AECT

ARMCortexA65AECT_CortexA76AECT CPU component. This model is written in C++.

ARMCortexA65AECT_CortexA76AECT contains the following CADI targets:

- ARM_Cortex-A65AE
- Cluster_ARM_Cortex-A65AE_Cortex-A76AE
- Subcluster_ARM_Cortex-A65AE
- Subcluster_ARM_Cortex-A76AE

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ARMCortexA65AECT_CortexA76AECT contains the following MTI components:

- *4.16 ARM_Cortex-A65AE - trace* on page 4-1329
- *4.22 ARM_Cortex-A76AE - trace* on page 4-1556
- *4.48 ARMv8Cluster - trace* on page 4-2401
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402
- *4.68 DSU - trace* on page 4-2463
- *4.73 GICv3CPUInterfaceDecoder - trace* on page 4-2489
- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.108 PVCache - trace* on page 4-2761
- *4.114 TLB - trace* on page 4-2834

About ARMCortexA65AECT_CortexA76AECT

For information about the cores in this model, see:

- *3.4.20 ARMCortexA65AECT* on page 3-384.
- *3.4.26 ARMCortexA76AECT* on page 3-444.

Table 3-136 Ports

Name	Protocol	Type	Description
AENDMP	*Value_64* on page 2-85	Slave	DynamIQ port to obtain end address of valid peripheral address range (exclusive).
ASTARTMP	*Value_64* on page 2-85	Slave	DynamIQ port to obtain start address of valid peripheral address range (inclusive).
CNTHPIRQ[12]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTHVIRQ[12]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTPNSIRQ[12]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTPSIRQ[12]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTVIRQ[12]	*Signal* on page 2-85	Master	Timer signals to SOC.
acp_s	PVBus	Slave	AXI ACP slave port
broadcastatomic	*Signal* on page 2-85	Slave	CHI defined pins.
broadcastcachemaint	*Signal* on page 2-85	Slave	ACE defined pins.
broadcastouter	*Signal* on page 2-85	Slave	ACE defined pins.
broadcastpersist	*Signal* on page 2-85	Slave	CHI defined pins.
cfgend[12]	*Signal* on page 2-85	Slave	This signal if for EE bit initialisation.
cfgte[12]	*Signal* on page 2-85	Slave	This signal provides default exception handling state.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-136 Ports (continued)

Name	Protocol	Type	Description
clk_in	ClockSignal	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the cluster level components run e.g cluster level timers, caches and pmu.
clusterid	*Value* on page 2-85	Slave	The port reads the value in CPU ID register field, bits[11:8] of the MPIDR.
clusterpmuirq	*Signal* on page 2-85	Master	DynamIQ pmu irq
cntvalueb	*CounterInterface* on page 2-84	Slave	Interface to SoC level counter module.
commirq[12]	*Signal* on page 2-85	Master	Interrupt signal from debug communications channel.
core_clk_in[8]	ClockSignal	Slave	The clock signal connected to the core_clk_in port is used to determine the rate at which each core executes instructions.
cpuporeset[8]	*Signal* on page 2-85	Slave	CPU power on reset. Initializes all the processor logic, including debug logic.
cryptodisable[12]	*Signal* on page 2-85	Slave	Disable cryptography extensions after reset.
cti[12]	*v8EmbeddedCrossTrigger_controlprotocol* on page 2-84	Master	Cross trigger matrix port.
ctidbgirq[12]	*Signal* on page 2-85	Master	Cross Trigger Interface (CTI) interrupt trigger output.
dbgen	*Signal* on page 2-85	Slave	External debug interface.
dbgnopwrdwn[12]	*Signal* on page 2-85	Master	No power-down request.
dbgpwrupreq[12]	*Signal* on page 2-85	Master	Debug power up request.
dev_debug_s	PVBus	Slave	External debug interface.
event	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE.
fiq[12]	*Signal* on page 2-85	Slave	This signal drives the CPUs fast-interrupt handling.
gicv3_redistributor_s[12]	*GICv3Comms* on page 2-84	Slave	GICv3 AXI-stream port.
irq[12]	*Signal* on page 2-85	Slave	This signal drives the CPUs interrupt handling.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-136 Ports (continued)

Name	Protocol	Type	Description
memorymapped_debug_s	PVBus	Slave	External debug interface.
niden	*Signal* on page 2-85	Slave	External debug interface.
pchannel_cluster	*PChannel* on page 2-80	Slave	PChannel for cluster.
pchannel_core[8]	*PChannel* on page 2-80	Slave	PChannels for cores
pmuirq[12]	*Signal* on page 2-85	Master	Interrupt signal from performance monitoring unit.
presetdbg	*Signal* on page 2-85	Slave	Initialize the shared debug APB, Cross Trigger Interface (CTI), and Cross Trigger Matrix (CTM) logic.
pvbus_m0	PVBus	Master	The core will generate bus requests on this port.
pvbus_periph_m	PVBus	Master	The core can generate peripheral bus request on this port.
rei[12]	*Signal* on page 2-85	Slave	Per core RAM Error Interrupt.
reset[8]	*Signal* on page 2-85	Slave	Raising this signal will put the core into reset mode.
rvbaraddr[12]	*Value_64* on page 2-85	Slave	Reset vector base address.
sci_m	SystemCoherencyInterface	Master	System coherency interface port, which is used to take the whole cluster into/out-of coherency domain
sei[12]	*Signal* on page 2-85	Slave	Per core System Error physical pins.
spiden	*Signal* on page 2-85	Slave	External debug interface.
spniden	*Signal* on page 2-85	Slave	External debug interface.
sporeset	*Signal* on page 2-85	Slave	A single cluster-wide power on reset signal for all resettable registers in DynamIQ.
ticks[12]	*InstructionCount* on page 2-84	Master	This port should be connected to one of the two ticks ports on a 'visualisation' component, in order to display a running instruction count.
vcpumntirq[12]	*Signal* on page 2-85	Master	Interrupt signal for virtual CPU maintenance IRQ.
vfiq[12]	*Signal* on page 2-85	Slave	Virtualised FIQ.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-136 Ports (continued)

Name	Protocol	Type	Description
vinithi[12]	*Signal* on page 2-85	Slave	This signal controls of the location of the exception vectors at reset.
virq[12]	*Signal* on page 2-85	Slave	Virtualised IRQ.
virtio_s	PVBus	Slave	The virtio coherent port, hooks directly into the L3 system and becomes coherent (assuming attributes are set correctly).
vsei[12]	*Signal* on page 2-85	Slave	Per core virtual System Error physical pins.

Table 3-137 Parameters for Subcluster_ARM_Cortex-A65AE

Name	Type	Default value	Description
subcluster0.NUM_CORES	int	0x2	Number of cores per cluster
subcluster0.cpi_div	int	0x1	Divider for calculating CPI (Cycles Per Instruction)
subcluster0.cpi_mul	int	0x1	Multiplier for calculating CPI (Cycles Per Instruction)
subcluster0.dcache-hit_latency	int	0x0	L1 D-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
subcluster0.dcache-maintenance_latency	int	0x0	L1 D-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
subcluster0.dcache-miss_latency	int	0x0	L1 D-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
subcluster0.dcache-prefetch_enabled	bool	0x0	Enable simulation of data cache prefetching. This is only used when dcache-state_modelled=true
subcluster0.dcache-read_access_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per access (of size dcache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
subcluster0.dcache-read_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per byte accessed.dcache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
subcluster0.dcache-size	int	0x10000	L1 D-Cache size in bytes.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-137 Parameters for Subcluster_ARM_Cortex-A65AE (continued)

Name	Type	Default value	Description
subcluster0.dcache-snoop_data_transfer_latency	int	0x0	L1 D-Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
subcluster0.dcache-write_access_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per access (of size dcache-write_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-write_latency is set. This is only used when dcache-state_modelled=true.
subcluster0.dcache-write_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per byte accessed. dcache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
subcluster0.has_dot_product	int	0x2	Possible values of this parameter are: - 1, feature not implemented. - 2, feature is implemented.
subcluster0.icache-hit_latency	int	0x0	L1 I-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when icache-state_modelled=true.
subcluster0.icache-maintenance_latency	int	0x0	L1 I-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when icache-state_modelled=true.
subcluster0.icache-miss_latency	int	0x0	L1 I-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when icache-state_modelled=true.
subcluster0.icache-prefetch_enabled	bool	0x0	Enable simulation of instruction cache prefetching. This is only used when icache-state_modelled=true.
subcluster0.icache-read_access_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per access (of size icache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if icache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when icache-state_modelled=true.
subcluster0.icache-read_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per byte accessed.icache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when icache-state_modelled=true.
subcluster0.icache-size	int	0x10000	L1 I-Cache size in bytes.
subcluster0.ptw_latency	int	0x0	Page table walker latency for TA (Timing Annotation), expressed in simulation ticks

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-137 Parameters for Subcluster_ARM_Cortex-A65AE (continued)

Name	Type	Default value	Description
subcluster0.tlb_latency	int	0x0	TLB latency for TA (Timing Annotation), expressed in simulation ticks
subcluster0.walk_cache_latency	int	0x0	Walk cache latency for TA (Timing Annotation), expressed in simulation ticks

Table 3-138 Parameters for ARM_Cortex-A65AE

Name	Type	Default value	Description
cpu0.thread0.CRYPTODISABLE	bool	0x0	Disable cryptographic features.
cpu0.thread0.enable_single_thread_at_reset	bool	0x0	Enable single thread after reset and keep other thread in reset
cpu0.thread0.enable_trace_special_hlt_imm16	bool	0x0	Enable usage of parameter trace_special_hlt_imm16
cpu0.thread0.l2cache-hit_latency	int	0x0	L2 Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
cpu0.thread0.l2cache-maintenance_latency	int	0x0	L2 Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
cpu0.thread0.l2cache-miss_latency	int	0x0	L2 Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
cpu0.thread0.l2cache-read_access_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
cpu0.thread0.l2cache-read_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per byte accessed.l2cache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
cpu0.thread0.l2cache-size	int	0x40000	L2 Cache size in bytes.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-138 Parameters for ARM_Cortex-A65AE (continued)

Name	Type	Default value	Description
cpu0.thread0.l2cache-snoop_data_transfer_latency	int	0x0	L2 Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
cpu0.thread0.l2cache-snoop_issue_latency	int	0x0	L2 Cache timing annotation latency for snoop accesses issued by this cache in total ticks. This is only used when dcache-state_modelled=true.
cpu0.thread0.l2cache-write_access_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-write_latency is set. This is only used when dcache-state_modelled=true.
cpu0.thread0.l2cache-write_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per byte accessed. l2cache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
cpu0.thread0.max_code_cache_mb	int	0x100	Maximum size of the simulation code cache (MiB). For platforms with more than 2 cores this limit will be scaled down. (e.g 1/8 for 16 or more cores)
cpu0.thread0.min_sync_level	int	0x0	Force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)
cpu0.thread0.semihosting-A32_HLT	int	0xf000	A32 HLT number for semihosting calls.
cpu0.thread0.semihosting-A64_HLT	int	0xf000	A64 HLT number for semihosting calls.
cpu0.thread0.semihosting-ARM_SVC	int	0x123456	A32 SVC number for semihosting calls.
cpu0.thread0.semihosting-T32_HLT	int	0x3c	T32 HLT number for semihosting calls.
cpu0.thread0.semihosting-Thumb_SVC	int	0xab	T32 SVC number for semihosting calls.
cpu0.thread0.semihosting-cmd_line	string	""	Command line available to semihosting calls.
cpu0.thread0.semihosting-cwd	string	""	Base directory for semihosting file access.
cpu0.thread0.semihosting-enable	bool	0x1	Enable semihosting SVC/HLT traps.
cpu0.thread0.semihosting-heap_base	int	0x0	Virtual address of heap base.
cpu0.thread0.semihosting-heap_limit	int	0xf000000	Virtual address of top of heap.
cpu0.thread0.semihosting-stack_base	int	0x10000000	Virtual address of base of descending stack.
cpu0.thread0.semihosting-stack_limit	int	0xf000000	Virtual address of stack limit.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-138 Parameters for ARM_Cortex-A65AE (continued)

Name	Type	Default value	Description
cpu0.thread0.trace_special_hlt_imm16	int	0xf000	For this HLT number, IF enable_trace_special_hlt_imm16=true, skip performing usual HLT execution but call MTI trace if registered
cpu0.thread0.vfp-enable_at_reset	bool	0x0	Enable VFP in CPACR, CPPWR, NSACR at reset. Warning: Arm recommends going through the implementation's suggested VFP power-up sequence!
cpu0.thread0.vfp-present	bool	0x1	Set whether the model has VFP support

Table 3-139 Parameters for Cluster_ARM_Cortex-A65AE_Cortex-A76AE

Name	Type	Default value	Description
BROADCASTATOMIC	bool	0x1	Enable broadcasting of atomic operation. The broadcastatomic signal will override this value if used
BROADCASTCACHEMAINT	bool	0x1	Enable broadcasting of cache maintenance operations to downstream caches. The broadcastcachemaint signal will override this value if used.
BROADCASTOUTER	bool	0x1	Enable broadcasting of Outer Shareable transactions. The broadcastouter signal will override this value if used.
BROADCASTPERSIST	bool	0x1	Enable broadcasting of cache clean to the point of persistence operations. The broadcastpersist signal will override this value if used
CLUSTER_ID	int	0x0	Processor cluster ID value
GICDISABLE	bool	0x1	Disable the new style GICv3 CPU interface in each core model. Should be left enabled unless the platform contains a GICv3 distributor.
dcache-state_modelled	bool	0x0	Set whether D-cache has stateful implementation
default_opmode	int	0x4	Operating mode of DynamIQ coming out of reset. 0: SFONLY ON, 1: 1/4 CACHE ON, 2: 1/2 CACHE ON, 3: 3/4 CACHE ON, 4: FULL CACHE ON
diagnostics	bool	0x0	Enable DynamIQ diagnostic messages
enable_simulation_performance_optimizations	bool	0x1	With this option enabled, the model will run more quickly, but be less accurate to exact CPU behavior. The model will still be functionally accurate for software, but may increase differences seen between hardware behavior and model behavior for certain workloads (it changes the micro-architectural value of stage12_tlb_size parameter to 1024).

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-139 Parameters for Cluster_ARM_Cortex-A65AE_Cortex-A76AE (continued)

Name	Type	Default value	Description
has_peripheral_port	bool	0x0	If true, additional AXI peripheral port is configured.
icache-state_modelled	bool	0x0	Set whether I-cache has stateful implementation
l3cache-hit_latency	int	0x0	L3 Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
l3cache-maintenance_latency	int	0x0	L3 Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
l3cache-miss_latency	int	0x0	L3 Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
l3cache-read_access_latency	int	0x0	L3 Cache timing annotation latency for read accesses given in ticks per access (of size l3cache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l3cache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
l3cache-read_latency	int	0x0	L3 Cache timing annotation latency for read accesses given in ticks per byte accessed.l3cache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
l3cache-size	int	0x100000	L3 Cache size in bytes.
l3cache-snoop_data_transfer_latency	int	0x0	L3 Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
l3cache-snoop_issue_latency	int	0x0	L3 Cache timing annotation latency for snoop accesses issued by this cache in total ticks. This is only used when dcache-state_modelled=true.
l3cache-write_access_latency	int	0x0	L3 Cache timing annotation latency for write accesses given in ticks per access (of size l3cache-write_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l3cache-write_latency is set. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-139 Parameters for Cluster_ARM_Cortex-A65AE_Cortex-A76AE (continued)

Name	Type	Default value	Description
l3cache-write_latency	int	0x0	L3 Cache timing annotation latency for write accesses given in ticks per byte accessed. l3cache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
pchannel_treat_simreset_as_poreset	bool	0x0	Register core as ON state to cluster with simulation reset.
periph_address_end	int	0x0	End address for peripheral port address range exclusive(corresponds to AENDMP input signal).
periph_address_start	int	0x0	Start address for peripheral port address range inclusive(corresponds to ASTARTMP input signal).

Table 3-140 Parameters for Subcluster_ARM_Cortex-A76AE

Name	Type	Default value	Description
subcluster1.NUM_CORES	int	0x2	Number of cores per cluster
subcluster1.cpi_div	int	0x1	Divider for calculating CPI (Cycles Per Instruction)
subcluster1.cpi_mul	int	0x1	Multiplier for calculating CPI (Cycles Per Instruction)
subcluster1.dcache-hit_latency	int	0x0	L1 D-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
subcluster1.dcache-maintenance_latency	int	0x0	L1 D-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
subcluster1.dcache-miss_latency	int	0x0	L1 D-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
subcluster1.dcache-prefetch_enabled	bool	0x0	Enable simulation of data cache prefetching. This is only used when dcache-state_modelled=true
subcluster1.dcache-read_access_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per access (of size dcache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-140 Parameters for Subcluster_ARM_Cortex-A76AE (continued)

Name	Type	Default value	Description
subcluster1.dcache-read_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per byte accessed.dcache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
subcluster1.dcache-snoop_data_transfer_latency	int	0x0	L1 D-Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
subcluster1.dcache-write_access_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per access (of size dcache-write_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-write_latency is set. This is only used when dcache-state_modelled=true.
subcluster1.dcache-write_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per byte accessed. dcache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
subcluster1.ext_abort_device_read_is_sync	bool	0x0	Synchronous reporting of device-nGnRE read external aborts
subcluster1.ext_abort_device_write_is_sync	bool	0x0	Synchronous reporting of device-nGnRE write external aborts
subcluster1.ext_abort_so_read_is_sync	bool	0x0	Synchronous reporting of device-nGnRnE read external aborts
subcluster1.ext_abort_so_write_is_sync	bool	0x0	Synchronous reporting of device-nGnRnE write external aborts
subcluster1.has_dot_product	int	0x2	Possible values of this parameter are: - 1, feature not implemented. - 2, feature is implemented.
subcluster1.has_statistical_profiling	bool	0x1	Whether Statistical Based Profiling is implemented
subcluster1.icache-hit_latency	int	0x0	L1 I-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when icache-state_modelled=true.
subcluster1.icache-maintenance_latency	int	0x0	L1 I-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when icache-state_modelled=true.
subcluster1.icache-miss_latency	int	0x0	L1 I-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when icache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-140 Parameters for Subcluster_ARM_Cortex-A76AE (continued)

Name	Type	Default value	Description
subcluster1.icache-prefetch_enabled	bool	0x0	Enable simulation of instruction cache prefetching. This is only used when icache-state_modelled=true.
subcluster1.icache-read_access_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per access (of size icache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if icache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when icache-state_modelled=true.
subcluster1.icache-read_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per byte accessed.icache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when icache-state_modelled=true.
subcluster1.ptw_latency	int	0x0	Page table walker latency for TA (Timing Annotation), expressed in simulation ticks
subcluster1.tlb_latency	int	0x0	TLB latency for TA (Timing Annotation), expressed in simulation ticks
subcluster1.treat-dcache-cmos-to-pou-as-nop	bool	0x0	Whether dcache invalidation to the point of unification is required for instruction to data coherence. true - Invalidate operations not required
subcluster1.walk_cache_latency	int	0x0	Walk cache latency for TA (Timing Annotation), expressed in simulation ticks

3.4.22 ARMCortexA65CT

ARMCortexA65CT CPU component. This model is written in C++ and models version r0p0 of the RTL.

ARMCortexA65CT contains the following CADI targets:

- ARM_Cortex-A65
- Cluster_ARM_Cortex-A65
- PVCache
- TlbCadi

ARMCortexA65CT contains the following MTI components:

- *4.15 ARM_Cortex-A65 - trace* on page 4-1291
- *4.48 ARMv8Cluster - trace* on page 4-2401
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402
- *4.68 DSU - trace* on page 4-2463
- *4.73 GICv3CPUInterfaceDecoder - trace* on page 4-2489
- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.108 PVCache - trace* on page 4-2761
- *4.114 TLB - trace* on page 4-2834

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

About ARMCortexA65CT

The model supports the following features:

- DynamIQ (FCM) system registers.
- Per-core L2 cache.
- A P-Channel for the cluster and for each core.
- Optional peripheral port.
- L3 cache partition.
- Per-core clock.
- `BROADCASTPERSIST` pin.

The model does not support the following features:

- `BROADCASTCACHEMAINTPOU` pin.
- `COREINSTRRET`, `COREINSTRRUN`, or `nPMBIRQ` signals.
- DynamIQ features that are irrelevant to the programmers' view simulation, for example:
  - Automatic CPU retention mode.
  - Level-3 Cache RAM retention.
- 256-bit wide output transactions.
- ECC and parity schemes are hardware-specific so are not supported.
- Self-test features (MBIST).
- Latency configuration.
- Snoop filtering.
- Debug power management signals `DBGPWRDUP` and `DBGRSTREQ` are not implemented, but `DBGPWRUPREQ` and `DBGNOPWRDWN` are implemented.
- Cache stashing capability.
- Each thread has its own L1Cache and L2Cache instead of a common cache that is shared by all threads of the core.
- Per-thread parameters are not yet supported but signals are implemented.

This component has a variable number of cores per cluster, specified using the `NUM_CORES` parameter. The following components also exist, with a fixed number of cores per cluster:

- ARMCortexA65x1CT.
- ARMCortexA65x2CT.
- ARMCortexA65x3CT.
- ARMCortexA65x4CT.

The per-core parameters are preceded by `cpu`*n*`.`, where *n* identifies the core (0-3).

Table 3-141 Ports

Name	Protocol	Type	Description
`AENDMP`	*Value_64* on page 2-85	Slave	Port to obtain end address of valid peripheral address range (exclusive).
`ASTARTMP`	*Value_64* on page 2-85	Slave	Port to obtain start address of valid peripheral address range (inclusive).
`CNTHPIRQ[16]`	*Signal* on page 2-85	Master	Timer signals to SOC.
`CNTHVIRQ[16]`	*Signal* on page 2-85	Master	Timer signals to SOC.
`CNTPNSIRQ[16]`	*Signal* on page 2-85	Master	Timer signals to SOC.
`CNTPSIRQ[16]`	*Signal* on page 2-85	Master	Timer signals to SOC.
`CNTVIRQ[16]`	*Signal* on page 2-85	Master	Timer signals to SOC.
`acp_s`	`PVBus`	Slave	AXI ACP slave port.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-141 Ports (continued)

Name	Protocol	Type	Description
broadcastatomic	*Signal* on page 2-85	Slave	CHI defined pins.
broadcastcachemaint	*Signal* on page 2-85	Slave	ACE defined pins.
broadcastouter	*Signal* on page 2-85	Slave	ACE defined pins.
broadcastpersist	*Signal* on page 2-85	Slave	CHI defined pins.
cfgend[16]	*Signal* on page 2-85	Slave	This signal if for EE bit initialisation.
cfgte[16]	*Signal* on page 2-85	Slave	This signal provides default exception handling state.
clk_in	ClockSignal	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the cluster level components run e.g cluster level timers, caches and pmu.
clusterid	*Value* on page 2-85	Slave	The port reads the value in CPU ID register field, bits[11:8] of the MPIDR.
clusterpmuirq	*Signal* on page 2-85	Master	DynamIQ pmu irq
cntvalueb	*CounterInterface* on page 2-84	Slave	Interface to SoC level counter module.
commirq[16]	*Signal* on page 2-85	Master	Interrupt signal from debug communications channel.
core_clk_in[8]	ClockSignal	Slave	The clock signal connected to the core_clk_in port is used to determine the rate at which each core executes instructions.
cpuporeset[8]	*Signal* on page 2-85	Slave	Power on reset. Initializes all the processor logic, including debug logic.
cryptodisable[16]	*Signal* on page 2-85	Slave	Disable cryptography extensions after reset.
cti[16]	*v8EmbeddedCrossTrigger_controlprotocol* on page 2-84	Master	Cross trigger matrix port.
ctidbgirq[16]	*Signal* on page 2-85	Master	Cross Trigger Interface (CTI) interrupt trigger output.
dbgen	*Signal* on page 2-85	Slave	External debug interface.
dbgnopwrdwn[16]	*Signal* on page 2-85	Master	These signals relate to core power down.
dbgpwrupreq[16]	*Signal* on page 2-85	Master	Debug power up request.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-141 Ports (continued)

Name	Protocol	Type	Description
dev_debug_s	PVBus	Slave	External debug interface.
event	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE.
fiq[16]	*Signal* on page 2-85	Slave	This signal drives the CPUs fast-interrupt handling.
gicv3_redistributor_s[16]	*GICv3Comms* on page 2-84	Slave	GICv3 AXI-stream port.
irq[16]	*Signal* on page 2-85	Slave	This signal drives the CPUs interrupt handling.
memorymapped_debug_s	PVBus	Slave	External debug interface.
niden	*Signal* on page 2-85	Slave	External debug interface.
pchannel_cluster	*PChannel* on page 2-80	Slave	PChannel for cluster.
pchannel_core[8]	*PChannel* on page 2-80	Slave	PChannels for cores
pmuirq[16]	*Signal* on page 2-85	Master	Interrupt signal from performance monitoring unit.
presetdbg	*Signal* on page 2-85	Slave	Initialize the shared debug APB, Cross Trigger Interface (CTI), and Cross Trigger Matrix (CTM) logic.
pvbus_m0	PVBus	Master	The core will generate bus requests on this port.
pvbus_periph_m	PVBus	Master	The core can generate peripheral bus request on this port.
rei[16]	*Signal* on page 2-85	Slave	Per core RAM Error Interrupt.
reset[8]	*Signal* on page 2-85	Slave	Raising this signal will put the core into reset mode.
rvbaraddr[16]	*Value_64* on page 2-85	Slave	Reset vector base address.
sci_m	SystemCoherencyInterface	Master	System coherency interface port, which is used to take the whole cluster into/out-of coherency domain
sei[16]	*Signal* on page 2-85	Slave	Per core virtual System Error physical pins.
spiden	*Signal* on page 2-85	Slave	External debug interface.
spniden	*Signal* on page 2-85	Slave	External debug interface.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-141 Ports (continued)

Name	Protocol	Type	Description
sporeset	*Signal* on page 2-85	Slave	A single cluster-wide power on reset signal for all resettable registers in DynamIQ.
ticks[16]	*InstructionCount* on page 2-84	Master	This port should be connected to one of the two ticks ports on a 'visualisation' component, in order to display a running instruction count.
vcpumntirq[16]	*Signal* on page 2-85	Master	Interrupt signal for virtual CPU maintenance IRQ.
vfiq[16]	*Signal* on page 2-85	Slave	Virtualised FIQ.
vinithi[16]	*Signal* on page 2-85	Slave	This signal controls of the location of the exception vectors at reset.
virq[16]	*Signal* on page 2-85	Slave	Virtualised IRQ.
virtio_s	PVBus	Slave	The virtio coherent port, hooks directly into the L2 system and becomes coherent (assuming attributes are set correctly).
vsei[16]	*Signal* on page 2-85	Slave	Per core virtual System Error physical pins.

Table 3-142 Parameters for Cluster_ARM_Cortex-A65

Name	Type	Default value	Description
BROADCASTATOMIC	bool	0x1	Enable broadcasting of atomic operation. The broadcastatomic signal will override this value if used
BROADCASTCACHEMAINT	bool	0x1	Enable broadcasting of cache maintenance operations to downstream caches. The broadcastcachemaint signal will override this value if used.
BROADCASTOUTER	bool	0x1	Enable broadcasting of Outer Shareable transactions. The broadcastouter signal will override this value if used.
BROADCASTPERSIST	bool	0x1	Enable broadcasting of cache clean to the point of persistence operations. The broadcastpersist signal will override this value if used
CLUSTER_ID	int	0x0	Processor cluster ID value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-142 Parameters for Cluster_ARM_Cortex-A65 (continued)

Name	Type	Default value	Description
GICDISABLE	bool	0x1	Disable the new style GICv3 CPU interface in each core model. Should be left enabled unless the platform contains a GICv3 distributor.
cluster_patch_level	int	0x0	Cosmetic change to patch number in CLUSTERIDR. Corresponds to the Y in rXpY.
cluster_revision_number	int	0x0	Cosmetic change to revision number in CLUSTERIDR, Corresponds to the X in rXpY.
cpi_div	int	0x1	Divider for calculating CPI (Cycles Per Instruction)
cpi_mul	int	0x1	Multiplier for calculating CPI (Cycles Per Instruction)
dcache-hit_latency	int	0x0	L1 D-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
dcache-maintenance_latency	int	0x0	L1 D-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
dcache-miss_latency	int	0x0	L1 D-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
dcache-prefetch_enabled	bool	0x0	Enable simulation of data cache prefetching. This is only used when dcache-state_modelled=true
dcache-read_access_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per access (of size dcache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
dcache-read_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per byte accessed.dcache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
dcache-size	int	0x10000	L1 D-Cache size in bytes.
dcache-snoop_data_transfer_latency	int	0x0	L1 D-Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-142 Parameters for Cluster_ARM_Cortex-A65 (continued)

Name	Type	Default value	Description
dcache-state_modelled	bool	0x0	Set whether D-cache has stateful implementation
dcache-write_access_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per access (of size dcache-write_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-write_latency is set. This is only used when dcache-state_modelled=true.
dcache-write_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per byte accessed. dcache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
default_opmode	int	0x4	Operating mode of DynamIQ coming out of reset. 0: SFONLY ON, 1: 1/4 CACHE ON, 2: 1/2 CACHE ON, 3: 3/4 CACHE ON, 4: FULL CACHE ON
diagnostics	bool	0x0	Enable DynamIQ diagnostic messages
enable_simulation_performance_optimizations	bool	0x1	With this option enabled, the model will run more quickly, but be less accurate to exact CPU behavior. The model will still be functionally accurate for software, but may increase differences seen between hardware behavior and model behavior for certain workloads (it changes the micro-architectural value of stage12_tlb_size parameter to 1024).
has_dot_product	int	0x2	Possible values of this parameter are: - 1, feature not implemented. - 2, feature is implemented.
has_peripheral_port	bool	0x0	If true, additional AXI peripheral port is configured.
icache-hit_latency	int	0x0	L1 I-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when icache-state_modelled=true.
icache-maintenance_latency	int	0x0	L1 I-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when icache-state_modelled=true.
icache-miss_latency	int	0x0	L1 I-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when icache-state_modelled=true.
icache-prefetch_enabled	bool	0x0	Enable simulation of instruction cache prefetching. This is only used when icache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-142 Parameters for Cluster_ARM_Cortex-A65 (continued)

Name	Type	Default value	Description
icache-read_access_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per access (of size icache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if icache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when icache-state_modelled=true.
icache-read_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per byte accessed.icache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when icache-state_modelled=true.
icache-size	int	0x10000	L1 I-Cache size in bytes.
icache-state_modelled	bool	0x0	Set whether I-cache has stateful implementation
l3cache-hit_latency	int	0x0	L3 Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
l3cache-maintenance_latency	int	0x0	L3 Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
l3cache-miss_latency	int	0x0	L3 Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
l3cache-read_access_latency	int	0x0	L3 Cache timing annotation latency for read accesses given in ticks per access (of size l3cache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l3cache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
l3cache-read_latency	int	0x0	L3 Cache timing annotation latency for read accesses given in ticks per byte accessed.l3cache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-142 Parameters for Cluster_ARM_Cortex-A65 (continued)

Name	Type	Default value	Description
l3cache-size	int	0x400000	L3 Cache size in bytes.
l3cache-snoop_data_transfer_latency	int	0x0	L3 Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
l3cache-snoop_issue_latency	int	0x0	L3 Cache timing annotation latency for snoop accesses issued by this cache in total ticks. This is only used when dcache-state_modelled=true.
l3cache-write_access_latency	int	0x0	L3 Cache timing annotation latency for write accesses given in ticks per access (of size l3cache-write_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l3cache-write_latency is set. This is only used when dcache-state_modelled=true.
l3cache-write_latency	int	0x0	L3 Cache timing annotation latency for write accesses given in ticks per byte accessed. l3cache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
pchannel_treat_simreset_as_poreset	bool	0x0	Register core as ON state to cluster with simulation reset.
periph_address_end	int	0x0	End address for peripheral port address range exclusive(corresponds to AENDMP input signal).
periph_address_start	int	0x0	Start address for peripheral port address range inclusive(corresponds to ASTARTMP input signal).
ptw_latency	int	0x0	Page table walker latency for TA (Timing Annotation), expressed in simulation ticks
tlb_latency	int	0x0	TLB latency for TA (Timing Annotation), expressed in simulation ticks
walk_cache_latency	int	0x0	Walk cache latency for TA (Timing Annotation), expressed in simulation ticks

Table 3-143 Parameters for ARM_Cortex-A65

Name	Type	Default value	Description
cpu0.thread0.CRYPTODISABLE	bool	0x0	Disable cryptographic features.
cpu0.thread0.enable_single_thread_at_reset	bool	0x0	Enable single thread after reset and keep other thread in reset
cpu0.thread0.enable_trace_special_hlt_imm16	bool	0x0	Enable usage of parameter trace_special_hlt_imm16

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-143 Parameters for ARM_Cortex-A65 (continued)

Name	Type	Default value	Description
cpu0.thread0.l2cache-hit_latency	int	0x0	L2 Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
cpu0.thread0.l2cache-maintenance_latency	int	0x0	L2 Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
cpu0.thread0.l2cache-miss_latency	int	0x0	L2 Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
cpu0.thread0.l2cache-read_access_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
cpu0.thread0.l2cache-read_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per byte accessed.l2cache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
cpu0.thread0.l2cache-size	int	0x40000	L2 Cache size in bytes.
cpu0.thread0.l2cache-snoop_data_transfer_latency	int	0x0	L2 Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
cpu0.thread0.l2cache-snoop_issue_latency	int	0x0	L2 Cache timing annotation latency for snoop accesses issued by this cache in total ticks. This is only used when dcache-state_modelled=true.
cpu0.thread0.l2cache-write_access_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-write_latency is set. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-143 Parameters for ARM_Cortex-A65 (continued)

Name	Type	Default value	Description
cpu0.thread0.l2cache-write_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per byte accessed. l2cache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
cpu0.thread0.max_code_cache_mb	int	0x100	Maximum size of the simulation code cache (MiB). For platforms with more than 2 cores this limit will be scaled down. (e.g 1/8 for 16 or more cores)
cpu0.thread0.min_sync_level	int	0x0	Force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)
cpu0.thread0.semihosting-A32_HLT	int	0xf000	A32 HLT number for semihosting calls.
cpu0.thread0.semihosting-A64_HLT	int	0xf000	A64 HLT number for semihosting calls.
cpu0.thread0.semihosting-ARM_SVC	int	0x123456	A32 SVC number for semihosting calls.
cpu0.thread0.semihosting-T32_HLT	int	0x3c	T32 HLT number for semihosting calls.
cpu0.thread0.semihosting-Thumb_SVC	int	0xab	T32 SVC number for semihosting calls.
cpu0.thread0.semihosting-cmd_line	string	""	Command line available to semihosting calls.
cpu0.thread0.semihosting-cwd	string	""	Base directory for semihosting file access.
cpu0.thread0.semihosting-enable	bool	0x1	Enable semihosting SVC/HLT traps.
cpu0.thread0.semihosting-heap_base	int	0x0	Virtual address of heap base.
cpu0.thread0.semihosting-heap_limit	int	0xf000000	Virtual address of top of heap.
cpu0.thread0.semihosting-stack_base	int	0x10000000	Virtual address of base of descending stack.
cpu0.thread0.semihosting-stack_limit	int	0xf000000	Virtual address of stack limit.
cpu0.thread0.trace_special_hlt_imm16	int	0xf000	For this HLT number, IF enable_trace_special_hlt_imm16=true, skip performing usual HLT execution but call MTI trace if registered
cpu0.thread0.vfp-enable_at_reset	bool	0x0	Enable VFP in CPACR, CPPWR, NSACR at reset. Warning: Arm recommends going through the implementation's suggested VFP power-up sequence!
cpu0.thread0.vfp-present	bool	0x1	Set whether the model has VFP support

3.4.23 ARMCortexA72x1CT

ARMCortexA72x1CT CPU component. This model is written in C++ and models version r0p0 of the RTL.

ARMCortexA72x1CT contains the following CADI targets:

- ARM_Cortex-A72
- Cluster_ARM_Cortex-A72

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

- PVCache
- TlbCadi

ARMCortexA72x1CT contains the following MTI components:

- *4.18 ARM_Cortex-A72 - trace* on page 4-1403
- *4.48 ARMv8Cluster - trace* on page 4-2401
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402
- *4.68 DSU - trace* on page 4-2463
- *4.73 GICv3CPUInterfaceDecoder - trace* on page 4-2489
- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.108 PVCache - trace* on page 4-2761
- *4.114 TLB - trace* on page 4-2834

About ARMCortexA72xnCT

- The following components also exist:
  - ARMCortexA72x2CT.
  - ARMCortexA72x3CT.
  - ARMCortexA72x4CT.

  The per-core parameters are preceded by `cpun.`, where *n* identifies the core (0-3).
- The cache latency cluster parameters are only effective when you enable cache-state modeling.
- Timing annotation for transactions downstream of the cache and TLB models propagates through the cache and TLB models.
- ECC and parity schemes are hardware-specific so are not supported.

Table 3-144 Ports

Name	Protocol	Type	Description
CNTHPIRQ[4]	*Signal* on page 2-85	Master	The per-EL counter signal.
CNTPNSIRQ[4]	*Signal* on page 2-85	Master	The per-EL counter signal.
CNTPSIRQ[4]	*Signal* on page 2-85	Master	The per-EL counter signal.
CNTVIRQ[4]	*Signal* on page 2-85	Master	The per-EL counter signal.
aa64naa32[4]	*Signal* on page 2-85	Slave	Register width after reset.
acp_s	PVBus	Slave	AXI ACP slave port.
broadcastcachemaint	*Signal* on page 2-85	Slave	Enable broadcasting of cache maintenance operations to downstream caches.
broadcastinner	*Signal* on page 2-85	Slave	Enable broadcasting of Inner Shareable transactions.
broadcastouter	*Signal* on page 2-85	Slave	Enable broadcasting of Outer Shareable transactions.
cfgend[4]	*Signal* on page 2-85	Slave	This signal if for EE bit initialisation.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-144 Ports (continued)

Name	Protocol	Type	Description
cfgsdisable	*Signal* on page 2-85	Slave	This signal disables write access to some secure Interrupt Controller registers.
cfgte[4]	*Signal* on page 2-85	Slave	This signal provides default exception handling state.
clk_in	ClockSignal	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the core executes instructions.
clrexmonack	*Signal* on page 2-85	Master	Acknowledge handshake signal for the clrexmonreq signal
clrexmonreq	*Signal* on page 2-85	Slave	Signals the clearing of an external global exclusive monitor
clusterid	*Value* on page 2-85	Slave	The port reads the value in CPU ID register field, bits[11:8] of the MPIDR.
cntvalueb	*CounterInterface* on page 2-84	Slave	Interface to SoC level counter module
commirq[4]	*Signal* on page 2-85	Master	Interrupt signal from debug communications channel.
commrx[4]	*Signal* on page 2-85	Master	Receive portion of Data Transfer Register full.
commtx[4]	*Signal* on page 2-85	Master	Transmit portion of Data Transfer Register empty.
cp15sdisable[4]	*Signal* on page 2-85	Slave	This signal disables write access to some system control processor registers.
cpuporeset[4]	*Signal* on page 2-85	Slave	Power on reset. Initializes all the processor logic, including debug logic.
cryptodisable[4]	*Signal* on page 2-85	Slave	Disable cryptography extensions after reset.
cti[4]	*v8EmbeddedCrossTrigger_controlprotocol* on page 2-84	Master	Cross trigger matrix port.
ctidbgirq[4]	*Signal* on page 2-85	Master	Cross Trigger Interface (CTI) interrupt trigger output.
dbgack[4]	*Signal* on page 2-85	Master	External debug interface.
dbgen[4]	*Signal* on page 2-85	Slave	External debug interface.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-144 Ports (continued)

Name	Protocol	Type	Description
dbgnopwrdwn[4]	*Signal* on page 2-85	Master	These signals relate to core power down.
dbgpwrupreq[4]	*Signal* on page 2-85	Master	These signals relate to core power down.
dev_debug_s	PVBus	Slave	External debug interface.
edbgrq[4]	*Signal* on page 2-85	Slave	External debug interface.
event	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE.
fiq[4]	*Signal* on page 2-85	Slave	This signal drives the CPUs fast-interrupt handling.
gicv3_redistributor_s[4]	*GICv3Comms* on page 2-84	Slave	GICv3 AXI-stream port.
irq[4]	*Signal* on page 2-85	Slave	This signal drives the CPUs interrupt handling.
l2flushdone	*Signal* on page 2-85	Master	Flush of L2 memory system complete.
l2flushreq	*Signal* on page 2-85	Slave	Request flush of L2 memory system.
l2reset	*Signal* on page 2-85	Slave	Reset the shared L2 memory system controller.
memorymapped_debug_s	PVBus	Slave	External debug interface.
niden[4]	*Signal* on page 2-85	Slave	External debug interface.
periphbase	*Value_64* on page 2-85	Slave	This port sets the base address of private peripheral region.
pmuirq[4]	*Signal* on page 2-85	Master	Interrupt signal from performance monitoring unit.
presetdbg	*Signal* on page 2-85	Slave	Initialize the shared debug APB, Cross Trigger Interface (CTI), and Cross Trigger Matrix (CTM) logic.
pvbus_m0	PVBus	Master	The core will generate bus requests on this port.
rei[4]	*Signal* on page 2-85	Slave	Individual processor RAM Error Interrupt signal input.
reset[4]	*Signal* on page 2-85	Slave	Raising this signal will put the core into reset mode.
romaddr	*Value_64* on page 2-85	Slave	Debug ROM base address.
romaddrv	*Signal* on page 2-85	Slave	Debug ROM base address valid.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-144 Ports (continued)

Name	Protocol	Type	Description
rvbaraddr[4]	*Value_64* on page 2-85	Slave	Reset vector base address.
sei[4]	*Signal* on page 2-85	Slave	Per core System Error physical pins.
smpen[4]	*Signal* on page 2-85	Master	This signals AMP or SMP mode for each core.
spiden[4]	*Signal* on page 2-85	Slave	External debug interface.
spniden[4]	*Signal* on page 2-85	Slave	External debug interface.
standbywfe[4]	*Signal* on page 2-85	Master	This signal indicates if a core is in WFE state.
standbywfi[4]	*Signal* on page 2-85	Master	This signal indicates if a core is in WFI state.
standbywfil2	*Signal* on page 2-85	Master	Indicate that all the individual processors and the L2 systems are in a WFI state.
ticks[4]	*InstructionCount* on page 2-84	Master	This port should be connected to one of the two ticks ports on a 'visualisation' component, in order to display a running instruction count.
vcpumntirq[4]	*Signal* on page 2-85	Master	Interrupt signal for virtual CPU maintenance IRQ.
vfiq[4]	*Signal* on page 2-85	Slave	Virtual FIQ
vinithi[4]	*Signal* on page 2-85	Slave	This signal controls of the location of the exception vectors at reset.
virq[4]	*Signal* on page 2-85	Slave	Virtual IRQ.
virtio_s	PVBus	Slave	The virtio coherent port, hooks directly into the L2 system and becomes coherent (assuming attributes are set correctly).
vsei[4]	*Signal* on page 2-85	Slave	Processor Virtual System Error Interrupt request.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-145 Parameters for Cluster_ARM_Cortex-A72

Name	Type	Default value	Description
BROADCASTCACHEMAINT	bool	0x1	Enable broadcasting of cache maintenance operations to downstream caches. The broadcastcachemaint signal will override this value if used.
BROADCASTINNER	bool	0x1	Enable broadcasting of Inner Shareable transactions. The broadcastinner signal will override this value if used.
BROADCASTOUTER	bool	0x1	Enable broadcasting of Outer Shareable transactions. The broadcastouter signal will override this value if used.
CLUSTER_ID	int	0x0	Processor cluster ID value
DBGROMADDR	int	0x0	Initialization value of DBGDRAR register. Bits[47:12] of this register specify the ROM table physical address.
DBGROMADDRV	bool	0x0	If true, set bits[1:0] of the CP15 DBGDRAR to indicate that the address is valid
GICDISABLE	bool	0x1	Disable the new style GICv3 CPU interface in each core model. Should be left enabled unless the platform contains a GICv3 distributor.
PERIPHBASE	int	0x13080000	Base address of peripheral memory space
bus_type	int	0x0	Cosmetic change that changes reset value of L2ACTLR register. 0, ACE. 1, CHI. 2, AXI
cpi_div	int	0x1	Divider for calculating CPI (Cycles Per Instruction)
cpi_mul	int	0x1	Multiplier for calculating CPI (Cycles Per Instruction)
dcache-hit_latency	int	0x0	L1 D-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
dcache-maintenance_latency	int	0x0	L1 D-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
dcache-miss_latency	int	0x0	L1 D-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
dcache-prefetch_enabled	bool	0x0	Enable simulation of data cache prefetching. This is only used when dcache-state_modelled=true

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-145 Parameters for Cluster_ARM_Cortex-A72 (continued)

Name	Type	Default value	Description
dcache-read_access_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per access (of size dcache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
dcache-read_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per byte accessed.dcache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
dcache-snoop_data_transfer_latency	int	0x0	L1 D-Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
dcache-state_modelled	bool	0x0	Set whether D-cache has stateful implementation
dcache-write_access_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per access (of size dcache-write_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-write_latency is set. This is only used when dcache-state_modelled=true.
dcache-write_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per byte accessed. dcache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
enable_simulation_performance_optimizations	bool	0x1	With this option enabled, the model will run more quickly, but be less accurate to exact CPU behavior. The model will still be functionally accurate for software, but may increase differences seen between hardware behavior and model behavior for certain workloads (it changes the micro-architectural value of stage12_tlb_size parameter to 1024).
icache-hit_latency	int	0x0	L1 I-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when icache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-145 Parameters for Cluster_ARM_Cortex-A72 (continued)

Name	Type	Default value	Description
icache-maintenance_latency	int	0x0	L1 I-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when icache-state_modelled=true.
icache-miss_latency	int	0x0	L1 I-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when icache-state_modelled=true.
icache-prefetch_enabled	bool	0x0	Enable simulation of instruction cache prefetching. This is only used when icache-state_modelled=true.
icache-read_access_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per access (of size icache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if icache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when icache-state_modelled=true.
icache-read_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per byte accessed.icache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when icache-state_modelled=true.
icache-state_modelled	bool	0x0	Set whether I-cache has stateful implementation
l2cache-hit_latency	int	0x0	L2 Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
l2cache-maintenance_latency	int	0x0	L2 Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
l2cache-miss_latency	int	0x0	L2 Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-145 Parameters for Cluster_ARM_Cortex-A72 (continued)

Name	Type	Default value	Description
l2cache-read_access_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
l2cache-read_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per byte accessed.l2cache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
l2cache-size	int	0x80000	L2 Cache size in bytes.
l2cache-snoop_data_transfer_latency	int	0x0	L2 Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
l2cache-snoop_issue_latency	int	0x0	L2 Cache timing annotation latency for snoop accesses issued by this cache in total ticks. This is only used when dcache-state_modelled=true.
l2cache-write_access_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-write_latency is set. This is only used when dcache-state_modelled=true.
l2cache-write_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per byte accessed. l2cache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
ptw_latency	int	0x0	Page table walker latency for TA (Timing Annotation), expressed in simulation ticks
tlb_latency	int	0x0	TLB latency for TA (Timing Annotation), expressed in simulation ticks
walk_cache_latency	int	0x0	Walk cache latency for TA (Timing Annotation), expressed in simulation ticks

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-146 Parameters for ARM_Cortex-A72

Name	Type	Default value	Description
cpu0.AA64nAA32	bool	0x1	Register width configuration at reset. 0, AArch32. 1, AArch64.
cpu0.CFGEND	bool	0x0	Endianness configuration at reset. 0, little endian. 1, big endian.
cpu0.CFGTE	bool	0x0	Instruction set state when resetting into AArch32. 0, A32. 1, T32.
cpu0.CP15SDISABLE	bool	0x0	Initialize to disable access to some CP15 registers
cpu0.CRYPTODISABLE	bool	0x0	Disable cryptographic features.
cpu0.RVBARADDR	int	0x0	Value of RVBAR_ELx register.
cpu0.VINITHI	bool	0x0	Reset value of SCTLR.V.
cpu0.enable_trace_special_hlt_imm16	bool	0x0	Enable usage of parameter trace_special_hlt_imm16
cpu0.max_code_cache_mb	int	0x100	Maximum size of the simulation code cache (MiB). For platforms with more than 2 cores this limit will be scaled down. (e.g 1/8 for 16 or more cores)
cpu0.min_sync_level	int	0x0	Force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)
cpu0.semihosting-A32_HLT	int	0xf000	A32 HLT number for semihosting calls.
cpu0.semihosting-A64_HLT	int	0xf000	A64 HLT number for semihosting calls.
cpu0.semihosting-ARM_SVC	int	0x123456	A32 SVC number for semihosting calls.
cpu0.semihosting-T32_HLT	int	0x3c	T32 HLT number for semihosting calls.
cpu0.semihosting-Thumb_SVC	int	0xab	T32 SVC number for semihosting calls.
cpu0.semihosting-cmd_line	string	""	Command line available to semihosting calls.
cpu0.semihosting-cwd	string	""	Base directory for semihosting file access.
cpu0.semihosting-enable	bool	0x1	Enable semihosting SVC/HLT traps.
cpu0.semihosting-heap_base	int	0x0	Virtual address of heap base.
cpu0.semihosting-heap_limit	int	0xf000000	Virtual address of top of heap.
cpu0.semihosting-stack_base	int	0x10000000	Virtual address of base of descending stack.
cpu0.semihosting-stack_limit	int	0xf000000	Virtual address of stack limit.
cpu0.trace_special_hlt_imm16	int	0xf000	For this HLT number, IF enable_trace_special_hlt_imm16=true, skip performing usual HLT execution but call MTI trace if registered
cpu0.vfp-enable_at_reset	bool	0x0	Enable VFP in CPACR, CPPWR, NSACR at reset. Warning: Arm recommends going through the implementation's suggested VFP power-up sequence!

3.4.24 ARMCortexA73x1CT

ARMCortexA73x1CT CPU component. This model is written in C++ and models version r0p2 of the RTL.

ARMCortexA73x1CT contains the following CADI targets:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

- ARM_Cortex-A73
- Cluster_ARM_Cortex-A73
- PVCache
- TlbCadi

ARMCortexA73x1CT contains the following MTI components:

- *4.19 ARM_Cortex-A73 - trace* on page 4-1441
- *4.48 ARMv8Cluster - trace* on page 4-2401
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402
- *4.68 DSU - trace* on page 4-2463
- *4.73 GICv3CPUInterfaceDecoder - trace* on page 4-2489
- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.108 PVCache - trace* on page 4-2761
- *4.114 TLB - trace* on page 4-2834

About ARMCortexA73x1CT

- The following components also exist:
  - ARMCortexA73x2CT.
  - ARMCortexA73x3CT.
  - ARMCortexA73x4CT.
- The per-core parameters are preceded by `cpun.`, where *n* identifies the core (0-3).
- This model supports the Armv8-A Cryptography Extensions, which require the Crypto plug-in to be loaded. See *Crypto plug-in* on page 5-2909 for details.
- If either L1 cache is stateful, then the L2 cache is stateful. This is controlled by the `dcache-state_modelled` and `icache-state_modelled` parameters.
- The `semihosting-cwd` parameter sets the CWD that is used for semihosting. The host operating system limits the maximum path length. The `semihosting-cwd` parameter does not provide any security. Software running on the model can access files outside this directory using relative paths containing `..` or using absolute paths.
- The system designer decides whether a debug APB ties the external debug view with other system views. In the model, use a `PVBusDecoder` to direct traffic to the correct port, `dev_debug_s` or `memorymapped_debug_s`.
- ECC support is hardware-specific so is not modeled.

Table 3-147 Ports

Name	Protocol	Type	Description
`CNTHPIRQ[4]`	*Signal* on page 2-85	Master	Timer signals to SOC.
`CNTPNSIRQ[4]`	*Signal* on page 2-85	Master	Timer signals to SOC.
`CNTPSIRQ[4]`	*Signal* on page 2-85	Master	Timer signals to SOC.
`CNTVIRQ[4]`	*Signal* on page 2-85	Master	Timer signals to SOC.
`aa64naa32[4]`	*Signal* on page 2-85	Slave	Register width after reset.
`acp_s`	`PVBus`	Slave	AXI ACP slave port.
`broadcastcachemaint`	*Signal* on page 2-85	Slave	ACE defined pins.
`broadcastinner`	*Signal* on page 2-85	Slave	ACE defined pins.
`broadcastouter`	*Signal* on page 2-85	Slave	ACE defined pins.
`cfgend[4]`	*Signal* on page 2-85	Slave	This signal if for EE bit initialisation.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-147 Ports (continued)

Name	Protocol	Type	Description
cfgsdisable	*Signal* on page 2-85	Slave	This signal disables write access to some secure Interrupt Controller registers.
cfgte[4]	*Signal* on page 2-85	Slave	This signal provides default exception handling state.
clk_in	ClockSignal	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the core executes instructions.
clrexmonack	*Signal* on page 2-85	Master	Acknowledge handshake signal for the clrexmonreq signal
clrexmonreq	*Signal* on page 2-85	Slave	Signals the clearing of an external global exclusive monitor
clusterid	*Value* on page 2-85	Slave	The port reads the value in CPU ID register field, bits[11:8] of the MPIDR.
cntvalueb	*CounterInterface* on page 2-84	Slave	Interface to SoC level counter module.
commirq[4]	*Signal* on page 2-85	Master	Interrupt signal from debug communications channel.
commrx[4]	*Signal* on page 2-85	Master	Receive portion of Data Transfer Register full.
commtx[4]	*Signal* on page 2-85	Master	Transmit portion of Data Transfer Register empty.
cp15sdisable[4]	*Signal* on page 2-85	Slave	This signal disables write access to some system control processor registers.
cpuporeset[4]	*Signal* on page 2-85	Slave	CPU power on reset.
cryptodisable[4]	*Signal* on page 2-85	Slave	Disable cryptography extensions after reset.
cti[4]	*v8EmbeddedCrossTrigger_controlprotocol* on page 2-84	Master	Cross trigger matrix port.
ctidbgirq[4]	*Signal* on page 2-85	Master	Cross trigger matrix port.
dbgack[4]	*Signal* on page 2-85	Master	External debug interface.
dbgen[4]	*Signal* on page 2-85	Slave	External debug interface.
dbgnopwrdwn[4]	*Signal* on page 2-85	Master	No power-down request.
dbgpwrupreq[4]	*Signal* on page 2-85	Master	Processor powerup request.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-147 Ports (continued)

Name	Protocol	Type	Description
dev_debug_s	PVBus	Slave	External debug interface.
edbgrq[4]	*Signal* on page 2-85	Slave	External debug interface.
event	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE.
fiq[4]	*Signal* on page 2-85	Slave	This signal drives the CPUs fast-interrupt handling.
gicv3_redistributor_s[4]	*GICv3Comms* on page 2-84	Slave	GICv3 AXI-stream port.
irq[4]	*Signal* on page 2-85	Slave	This signal drives the CPUs interrupt handling.
l2flushdone	*Signal* on page 2-85	Master	Flush of L2 memory system complete.
l2flushreq	*Signal* on page 2-85	Slave	Request flush of L2 memory system.
l2reset	*Signal* on page 2-85	Slave	Level2 reset.
memorymapped_debug_s	PVBus	Slave	External debug interface.
niden[4]	*Signal* on page 2-85	Slave	External debug interface.
periphbase	*Value_64* on page 2-85	Slave	This port sets the base address of private peripheral region.
pmuirq[4]	*Signal* on page 2-85	Master	Interrupt signal from performance monitoring unit.
presetdbg	*Signal* on page 2-85	Slave	Debug reset.
pvbus_m0	PVBus	Master	The core will generate bus requests on this port.
rei[4]	*Signal* on page 2-85	Slave	Per core RAM Error Interrupt
reset[4]	*Signal* on page 2-85	Slave	Reset.
romaddr	*Value_64* on page 2-85	Slave	Debug ROM base address.
romaddrv	*Signal* on page 2-85	Slave	Debug ROM base address valid.
rvbaraddr[4]	*Value_64* on page 2-85	Slave	Reset vector base address.
sei[4]	*Signal* on page 2-85	Slave	Per core System Error physical pins.
smpen[4]	*Signal* on page 2-85	Master	This signals AMP or SMP mode for each core.
spiden[4]	*Signal* on page 2-85	Slave	External debug interface.
spniden[4]	*Signal* on page 2-85	Slave	External debug interface.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-147 Ports (continued)

Name	Protocol	Type	Description
standbywfe[4]	*Signal* on page 2-85	Master	This signal indicates if a core is in WFE state.
standbywfi[4]	*Signal* on page 2-85	Master	This signal indicates if a core is in WFI state
standbywfil2	*Signal* on page 2-85	Master	This signal indicates all cores and L2 are idle and in low power state.
ticks[4]	*InstructionCount* on page 2-84	Master	This port should be connected to one of the two ticks ports on a 'visualisation' component, in order to display a running instruction count.
vcpumntirq[4]	*Signal* on page 2-85	Master	Interrupt signal for virtual CPU maintenance IRQ.
vfiq[4]	*Signal* on page 2-85	Slave	Virtualised FIQ.
vinithi[4]	*Signal* on page 2-85	Slave	This signal controls of the location of the exception vectors at reset.
virq[4]	*Signal* on page 2-85	Slave	Virtualised IRQ.
virtio_s	PVBus	Slave	The virtio coherent port, hooks directly into the L2 system and becomes coherent (assuming attributes are set correctly).
vsei[4]	*Signal* on page 2-85	Slave	Per core virtual System Error physical pins.

Table 3-148 Parameters for ARM_Cortex-A73

Name	Type	Default value	Description
cpu0.AA64nAA32	bool	0x1	Register width configuration at reset. 0, AArch32. 1, AArch64.
cpu0.CFGEND	bool	0x0	Endianness configuration at reset. 0, little endian. 1, big endian.
cpu0.CFGTE	bool	0x0	Instruction set state when resetting into AArch32. 0, A32. 1, T32.
cpu0.CP15SDISABLE	bool	0x0	Initialize to disable access to some CP15 registers
cpu0.CRYPTODISABLE	bool	0x0	Disable cryptographic features.
cpu0.RVBARADDR	int	0x0	Value of RVBAR_ELx register.
cpu0.SMPEN	bool	0x0	Enable broadcast messages necessary for correct SMP operation at reset.
cpu0.VINITHI	bool	0x0	Reset value of SCTLR.V.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-148 Parameters for ARM_Cortex-A73 (continued)

Name	Type	Default value	Description
cpu0.enable_trace_special_hlt_imm16	bool	0x0	Enable usage of parameter trace_special_hlt_imm16
cpu0.max_code_cache_mb	int	0x100	Maximum size of the simulation code cache (MiB). For platforms with more than 2 cores this limit will be scaled down. (e.g 1/8 for 16 or more cores)
cpu0.min_sync_level	int	0x0	Force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)
cpu0.semihosting-A32_HLT	int	0xf000	A32 HLT number for semihosting calls.
cpu0.semihosting-A64_HLT	int	0xf000	A64 HLT number for semihosting calls.
cpu0.semihosting-ARM_SVC	int	0x123456	A32 SVC number for semihosting calls.
cpu0.semihosting-T32_HLT	int	0x3c	T32 HLT number for semihosting calls.
cpu0.semihosting-Thumb_SVC	int	0xab	T32 SVC number for semihosting calls.
cpu0.semihosting-cmd_line	string	""	Command line available to semihosting calls.
cpu0.semihosting-cwd	string	""	Base directory for semihosting file access.
cpu0.semihosting-enable	bool	0x1	Enable semihosting SVC/HLT traps.
cpu0.semihosting-heap_base	int	0x0	Virtual address of heap base.
cpu0.semihosting-heap_limit	int	0xf000000	Virtual address of top of heap.
cpu0.semihosting-stack_base	int	0x10000000	Virtual address of base of descending stack.
cpu0.semihosting-stack_limit	int	0xf000000	Virtual address of stack limit.
cpu0.trace_special_hlt_imm16	int	0xf000	For this HLT number, IF enable_trace_special_hlt_imm16=true, skip performing usual HLT execution but call MTI trace if registered
cpu0.vfp-enable_at_reset	bool	0x0	Enable VFP in CPACR, CPPWR, NSACR at reset. Warning: Arm recommends going through the implementation's suggested VFP power-up sequence!

Table 3-149 Parameters for Cluster_ARM_Cortex-A73

Name	Type	Default value	Description
BROADCASTCACHEMAINT	bool	0x1	Enable broadcasting of cache maintenance operations to downstream caches. The broadcastcachemaint signal will override this value if used.
BROADCASTINNER	bool	0x1	Enable broadcasting of Inner Shareable transactions. The broadcastinner signal will override this value if used.
BROADCASTOUTER	bool	0x1	Enable broadcasting of Outer Shareable transactions. The broadcastouter signal will override this value if used.
CCSIDR-L1D_override	int	0x0	If nonzero, override the value presented in CCSIDR for L1D (this is cosmetic and does not affect cache behaviour).

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-149 Parameters for Cluster_ARM_Cortex-A73 (continued)

Name	Type	Default value	Description
CCSIDR-L1I_override	int	0x0	If nonzero, override the value presented in CCSIDR for L1I (this is cosmetic and does not affect cache behaviour).
CCSIDR-L2_override	int	0x0	If nonzero, override the value presented in CCSIDR for L2 (this is cosmetic and does not affect cache behaviour).
CLUSTER_ID	int	0x0	Processor cluster ID value
DBGROMADDR	int	0x0	Initialization value of DBGDRAR register. Bits[47:12] of this register specify the ROM table physical address.
DBGROMADDRV	bool	0x0	If true, set bits[1:0] of the CP15 DBGDRAR to indicate that the address is valid
GICDISABLE	bool	0x1	Disable the new style GICv3 CPU interface in each core model. Should be left enabled unless the platform contains a GICv3 distributor.
MIDR	int	0x411fd090	Value of MIDR_EL1 register.
PERIPHBASE	int	0x13080000	Base address of peripheral memory space
cpi_div	int	0x1	Divider for calculating CPI (Cycles Per Instruction)
cpi_mul	int	0x1	Multiplier for calculating CPI (Cycles Per Instruction)
dcache-hit_latency	int	0x0	L1 D-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
dcache-maintenance_latency	int	0x0	L1 D-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
dcache-miss_latency	int	0x0	L1 D-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
dcache-prefetch_enabled	bool	0x0	Enable simulation of data cache prefetching. This is only used when dcache-state_modelled=true
dcache-read_access_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per access (of size dcache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-149 Parameters for Cluster_ARM_Cortex-A73 (continued)

Name	Type	Default value	Description
dcache-read_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per byte accessed.dcache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
dcache-size	int	0x8000	L1 D-Cache size in bytes.
dcache-snoop_data_transfer_latency	int	0x0	L1 D-Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
dcache-state_modelled	bool	0x0	Set whether D-cache has stateful implementation
dcache-write_access_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per access (of size dcache-write_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-write_latency is set. This is only used when dcache-state_modelled=true.
dcache-write_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per byte accessed. dcache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
enable_simulation_performance_optimizations	bool	0x1	With this option enabled, the model will run more quickly, but be less accurate to exact CPU behavior. The model will still be functionally accurate for software, but may increase differences seen between hardware behavior and model behavior for certain workloads (it changes the micro-architectural value of stage12_tlb_size parameter to 1024).
icache-hit_latency	int	0x0	L1 I-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when icache-state_modelled=true.
icache-maintenance_latency	int	0x0	L1 I-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when icache-state_modelled=true.
icache-miss_latency	int	0x0	L1 I-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when icache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-149 Parameters for Cluster_ARM_Cortex-A73 (continued)

Name	Type	Default value	Description
icache-prefetch_enabled	bool	0x0	Enable simulation of instruction cache prefetching. This is only used when icache-state_modelled=true.
icache-read_access_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per access (of size icache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if icache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when icache-state_modelled=true.
icache-read_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per byte accessed.icache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when icache-state_modelled=true.
icache-state_modelled	bool	0x0	Set whether I-cache has stateful implementation
l2cache-hit_latency	int	0x0	L2 Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
l2cache-maintenance_latency	int	0x0	L2 Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
l2cache-miss_latency	int	0x0	L2 Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
l2cache-read_access_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-149 Parameters for Cluster_ARM_Cortex-A73 (continued)

Name	Type	Default value	Description
`l2cache-read_latency`	`int`	`0x0`	L2 Cache timing annotation latency for read accesses given in ticks per byte accessed.l2cache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
`l2cache-size`	`int`	`0x80000`	L2 Cache size in bytes.
`l2cache-snoop_data_transfer_latency`	`int`	`0x0`	L2 Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
`l2cache-snoop_issue_latency`	`int`	`0x0`	L2 Cache timing annotation latency for snoop accesses issued by this cache in total ticks. This is only used when dcache-state_modelled=true.
`l2cache-write_access_latency`	`int`	`0x0`	L2 Cache timing annotation latency for write accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-write_latency is set. This is only used when dcache-state_modelled=true.
`l2cache-write_latency`	`int`	`0x0`	L2 Cache timing annotation latency for write accesses given in ticks per byte accessed. l2cache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
`ptw_latency`	`int`	`0x0`	Page table walker latency for TA (Timing Annotation), expressed in simulation ticks
`tlb_latency`	`int`	`0x0`	TLB latency for TA (Timing Annotation), expressed in simulation ticks
`walk_cache_latency`	`int`	`0x0`	Walk cache latency for TA (Timing Annotation), expressed in simulation ticks

3.4.25 ARMCortexA75CT

ARMCortexA75CT CPU component. This model is written in C++.

ARMCortexA75CT contains the following CADI targets:

- ARM_Cortex-A75
- Cluster_ARM_Cortex-A75
- PVCache
- TlbCadi

ARMCortexA75CT contains the following MTI components:

- *4.20 ARM_Cortex-A75 - trace* on page 4-1479
- *4.48 ARMv8Cluster - trace* on page 4-2401
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

- *4.68 DSU - trace* on page 4-2463
- *4.73 GICv3CPUInterfaceDecoder - trace* on page 4-2489
- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.108 PVCache - trace* on page 4-2761
- *4.114 TLB - trace* on page 4-2834

About ARMCortexA75CT

The model supports the following features:

- DynamIQ (FCM) system registers.
- Per-core L2 cache.
- A P-Channel for the cluster and for each core.
- Optional peripheral port.
- L3 cache partition.
- Per-core clock.

The model does not support the following features:

- `BROADCASTCACHEMAINTPOU` pin.
- `COREINSTRRET`, `COREINSTRRUN`, or `nPMBIRQ` signals.
- DynamIQ features that are irrelevant to the programmers' view simulation, for example:
  - Automatic CPU retention mode.
  - Level-3 Cache RAM retention.
- 256-bit wide output transactions.
- ECC and parity schemes are hardware-specific so are not supported.
- Self-test features (MBIST).
- Latency configuration.
- Snoop filtering.
- Debug power management signals `DBGPWRDUP` and `DBGRSTREQ` are not implemented, but `DBGPWRUPREQ` and `DBGNOPWRDWN` are implemented.
- Cache stashing capability.
- Dual ACE masters.

This model has a variable number of cores per cluster, specified using the `NUM_CORES` parameter. The following models also exist, with a fixed number of cores per cluster:

- ARMCortexA75x1CT.
- ARMCortexA75x2CT.
- ARMCortexA75x3CT.
- ARMCortexA75x4CT.

The per-core parameters are preceded by `cpu`*n*`.`, where *n* identifies the core (0-3).

Table 3-150 Ports

Name	Protocol	Type	Description
`AENDMP`	*Value_64* on page 2-85	Slave	Port to obtain end address of valid peripheral address range (exclusive).
`ASTARTMP`	*Value_64* on page 2-85	Slave	Port to obtain start address of valid peripheral address range (inclusive).
`CNTHPIRQ[4]`	*Signal* on page 2-85	Master	Timer signals to SOC.
`CNTHVIRQ[4]`	*Signal* on page 2-85	Master	Timer signals to SOC.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-150 Ports (continued)

Name	Protocol	Type	Description
CNTPNSIRQ[4]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTPSIRQ[4]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTVIRQ[4]	*Signal* on page 2-85	Master	Timer signals to SOC.
aa64naa32[4]	*Signal* on page 2-85	Slave	Register width after reset.
acp_s	PVBus	Slave	AXI ACP slave port
broadcastatomic	*Signal* on page 2-85	Slave	CHI defined pins.
broadcastcachemaint	*Signal* on page 2-85	Slave	ACE defined pins.
broadcastouter	*Signal* on page 2-85	Slave	ACE defined pins.
broadcastpersist	*Signal* on page 2-85	Slave	CHI defined pins.
cfgend[4]	*Signal* on page 2-85	Slave	This signal if for EE bit initialisation.
cfgsdisable	*Signal* on page 2-85	Slave	This signal disables write access to some secure Interrupt Controller registers.
cfgte[4]	*Signal* on page 2-85	Slave	This signal provides default exception handling state.
clk_in	ClockSignal	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the cluster level components run e.g cluster level timers, caches and pmu.
clusterid	*Value* on page 2-85	Slave	The port reads the value in CPU ID register field, bits[11:8] of the MPIDR.
clusterpmuirq	*Signal* on page 2-85	Master	DynamIQ pmu irq
cntvalueb	*CounterInterface* on page 2-84	Slave	Interface to SoC level counter module.
commirq[4]	*Signal* on page 2-85	Master	Interrupt signal from debug communications channel.
core_clk_in[4]	ClockSignal	Slave	The clock signal connected to the core_clk_in port is used to determine the rate at which each core executes instructions.
cp15sdisable[4]	*Signal* on page 2-85	Slave	This signal disables write access to some system control processor registers.
cpuporeset[4]	*Signal* on page 2-85	Slave	Power on reset. Initializes all the processor logic, including debug logic.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-150 Ports (continued)

Name	Protocol	Type	Description
cryptodisable[4]	*Signal* on page 2-85	Slave	Disable cryptography extensions after reset.
cti[4]	*v8EmbeddedCrossTrigger_controlprotocol* on page 2-84	Master	Cross trigger matrix port.
ctidbgirq[4]	*Signal* on page 2-85	Master	Cross Trigger Interface (CTI) interrupt trigger output.
dbgen[4]	*Signal* on page 2-85	Slave	External debug interface.
dbgnopwrdwn[4]	*Signal* on page 2-85	Master	No power-down request.
dbgpwrupreq[4]	*Signal* on page 2-85	Master	Debug power up request.
dev_debug_s	PVBus	Slave	External debug interface.
edbgrq[4]	*Signal* on page 2-85	Slave	External debug interface.
event	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE.
fiq[4]	*Signal* on page 2-85	Slave	This signal drives the CPUs fast-interrupt handling.
gicv3_redistributor_s[4]	*GICv3Comms* on page 2-84	Slave	GICv3 AXI-stream port.
irq[4]	*Signal* on page 2-85	Slave	This signal drives the CPUs interrupt handling.
memorymapped_debug_s	PVBus	Slave	External debug interface.
niden[4]	*Signal* on page 2-85	Slave	External debug interface.
pchannel_cluster	*PChannel* on page 2-80	Slave	PChannel for cluster.
pchannel_core[4]	*PChannel* on page 2-80	Slave	PChannels for cores
pmuirq[4]	*Signal* on page 2-85	Master	Interrupt signal from performance monitoring unit.
presetdbg	*Signal* on page 2-85	Slave	Initialize the shared debug APB, Cross Trigger Interface (CTI), and Cross Trigger Matrix (CTM) logic.
pvbus_m0	PVBus	Master	The core will generate bus requests on this port.
pvbus_periph_m	PVBus	Master	The core can generate peripheral bus request on this port.
rei[4]	*Signal* on page 2-85	Slave	Individual processor RAM Error Interrupt signal input.
reset[4]	*Signal* on page 2-85	Slave	Raising this signal will put the core into reset mode.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-150 Ports (continued)

Name	Protocol	Type	Description
rvbaraddr[4]	*Value_64* on page 2-85	Slave	Reset vector base address.
sci_m	SystemCoherencyInterface	Master	System coherency interface port, which is used to take the whole cluster into/out-of coherency domain
sei[4]	*Signal* on page 2-85	Slave	Per core virtual System Error physical pins.
spiden[4]	*Signal* on page 2-85	Slave	External debug interface.
spniden[4]	*Signal* on page 2-85	Slave	External debug interface.
sporeset	*Signal* on page 2-85	Slave	A single cluster-wide power on reset signal for all resettable registers in DynamIQ.
ticks[4]	*InstructionCount* on page 2-84	Master	This port should be connected to one of the two ticks ports on a 'visualisation' component, in order to display a running instruction count.
vcpumntirq[4]	*Signal* on page 2-85	Master	Interrupt signal for virtual CPU maintenance IRQ.
vfiq[4]	*Signal* on page 2-85	Slave	Virtual FIQ.
vinithi[4]	*Signal* on page 2-85	Slave	This signal controls of the location of the exception vectors at reset.
virq[4]	*Signal* on page 2-85	Slave	Virtual IRQ.
virtio_s	PVBus	Slave	The virtio coherent port, hooks directly into the L2 system and becomes coherent (assuming attributes are set correctly).
vsei[4]	*Signal* on page 2-85	Slave	Per core virtual System Error physical pins.

Table 3-151 Parameters for ARM_Cortex-A75

Name	Type	Default value	Description
cpu0.AA64nAA32	bool	0x1	Register width configuration at reset. 0, AArch32. 1, AArch64.
cpu0.CFGEND	bool	0x0	Endianness configuration at reset. 0, little endian. 1, big endian.
cpu0.CFGTE	bool	0x0	Instruction set state when resetting into AArch32. 0, A32. 1, T32.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-151 Parameters for ARM_Cortex-A75 (continued)

Name	Type	Default value	Description
cpu0.CP15SDISABLE	bool	0x0	Initialize to disable access to some CP15 registers
cpu0.CRYPTODISABLE	bool	0x0	Disable cryptographic features.
cpu0.RVBARADDR	int	0x0	Value of RVBAR_ELx register.
cpu0.VINITHI	bool	0x0	Reset value of SCTLR.V.
cpu0.enable_trace_special_hlt_imm16	bool	0x0	Enable usage of parameter trace_special_hlt_imm16
cpu0.l2cache-hit_latency	int	0x0	L2 Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
cpu0.l2cache-maintenance_latency	int	0x0	L2 Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
cpu0.l2cache-miss_latency	int	0x0	L2 Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
cpu0.l2cache-read_access_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
cpu0.l2cache-read_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per byte accessed.l2cache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
cpu0.l2cache-size	int	0x80000	L2 Cache size in bytes.
cpu0.l2cache-snoop_data_transfer_latency	int	0x0	L2 Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
cpu0.l2cache-snoop_issue_latency	int	0x0	L2 Cache timing annotation latency for snoop accesses issued by this cache in total ticks. This is only used when dcache-state_modelled=true.
cpu0.l2cache-write_access_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-write_latency is set. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-151 Parameters for ARM_Cortex-A75 (continued)

Name	Type	Default value	Description
cpu0.l2cache-write_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per byte accessed. l2cache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
cpu0.max_code_cache_mb	int	0x100	Maximum size of the simulation code cache (MiB). For platforms with more than 2 cores this limit will be scaled down. (e.g 1/8 for 16 or more cores)
cpu0.min_sync_level	int	0x0	Force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)
cpu0.semihosting-A32_HLT	int	0xf000	A32 HLT number for semihosting calls.
cpu0.semihosting-A64_HLT	int	0xf000	A64 HLT number for semihosting calls.
cpu0.semihosting-ARM_SVC	int	0x123456	A32 SVC number for semihosting calls.
cpu0.semihosting-T32_HLT	int	0x3c	T32 HLT number for semihosting calls.
cpu0.semihosting-Thumb_SVC	int	0xab	T32 SVC number for semihosting calls.
cpu0.semihosting-cmd_line	string	""	Command line available to semihosting calls.
cpu0.semihosting-cwd	string	""	Base directory for semihosting file access.
cpu0.semihosting-enable	bool	0x1	Enable semihosting SVC/HLT traps.
cpu0.semihosting-heap_base	int	0x0	Virtual address of heap base.
cpu0.semihosting-heap_limit	int	0xf000000	Virtual address of top of heap.
cpu0.semihosting-stack_base	int	0x10000000	Virtual address of base of descending stack.
cpu0.semihosting-stack_limit	int	0xf000000	Virtual address of stack limit.
cpu0.trace_special_hlt_imm16	int	0xf000	For this HLT number, IF enable_trace_special_hlt_imm16=true, skip performing usual HLT execution but call MTI trace if registered
cpu0.vfp-enable_at_reset	bool	0x0	Enable VFP in CPACR, CPPWR, NSACR at reset. Warning: Arm recommends going through the implementation's suggested VFP power-up sequence!

Table 3-152 Parameters for Cluster_ARM_Cortex-A75

Name	Type	Default value	Description
BROADCASTATOMIC	bool	0x1	Enable broadcasting of atomic operation. The broadcastatomic signal will override this value if used
BROADCASTCACHEMAINT	bool	0x1	Enable broadcasting of cache maintenance operations to downstream caches. The broadcastcachemaint signal will override this value if used.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-152 Parameters for Cluster_ARM_Cortex-A75 (continued)

Name	Type	Default value	Description
BROADCASTOUTER	bool	0x1	Enable broadcasting of Outer Shareable transactions. The broadcastouter signal will override this value if used.
BROADCASTPERSIST	bool	0x1	Enable broadcasting of cache clean to the point of persistence operations. The broadcastpersist signal will override this value if used
CCSIDR-L1D_override	int	0x0	If nonzero, override the value presented in CCSIDR for L1D (this is cosmetic and does not affect cache behaviour).
CCSIDR-L1I_override	int	0x0	If nonzero, override the value presented in CCSIDR for L1I (this is cosmetic and does not affect cache behaviour).
CCSIDR-L2_override	int	0x0	If nonzero, override the value presented in CCSIDR for L2 (this is cosmetic and does not affect cache behaviour).
CLUSTER_ID	int	0x0	Processor cluster ID value
GICDISABLE	bool	0x1	Disable the new style GICv3 CPU interface in each core model. Should be left enabled unless the platform contains a GICv3 distributor.
cpi_div	int	0x1	Divider for calculating CPI (Cycles Per Instruction)
cpi_mul	int	0x1	Multiplier for calculating CPI (Cycles Per Instruction)
dcache-hit_latency	int	0x0	L1 D-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
dcache-maintenance_latency	int	0x0	L1 D-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
dcache-miss_latency	int	0x0	L1 D-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
dcache-prefetch_enabled	bool	0x0	Enable simulation of data cache prefetching. This is only used when dcache-state_modelled=true
dcache-read_access_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per access (of size dcache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-152 Parameters for Cluster_ARM_Cortex-A75 (continued)

Name	Type	Default value	Description
dcache-read_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per byte accessed.dcache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
dcache-snoop_data_transfer_latency	int	0x0	L1 D-Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
dcache-state_modelled	bool	0x0	Set whether D-cache has stateful implementation
dcache-write_access_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per access (of size dcache-write_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-write_latency is set. This is only used when dcache-state_modelled=true.
dcache-write_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per byte accessed. dcache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
default_opmode	int	0x4	Operating mode of DynamIQ coming out of reset. 0: SFONLY ON, 1: 1/4 CACHE ON, 2: 1/2 CACHE ON, 3: 3/4 CACHE ON, 4: FULL CACHE ON
diagnostics	bool	0x0	Enable DynamIQ diagnostic messages
enable_simulation_performance_optimizations	bool	0x1	With this option enabled, the model will run more quickly, but be less accurate to exact CPU behavior. The model will still be functionally accurate for software, but may increase differences seen between hardware behavior and model behavior for certain workloads (it changes the micro-architectural value of stage12_tlb_size parameter to 1024).
has_peripheral_port	bool	0x0	If true, additional AXI peripheral port is configured.
icache-hit_latency	int	0x0	L1 I-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when icache-state_modelled=true.
icache-maintenance_latency	int	0x0	L1 I-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when icache-state_modelled=true.
icache-miss_latency	int	0x0	L1 I-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when icache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-152 Parameters for Cluster_ARM_Cortex-A75 (continued)

Name	Type	Default value	Description
icache-prefetch_enabled	bool	0x0	Enable simulation of instruction cache prefetching. This is only used when icache-state_modelled=true.
icache-read_access_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per access (of size icache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if icache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when icache-state_modelled=true.
icache-read_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per byte accessed.icache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when icache-state_modelled=true.
icache-state_modelled	bool	0x0	Set whether I-cache has stateful implementation
l3cache-hit_latency	int	0x0	L3 Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
l3cache-maintenance_latency	int	0x0	L3 Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
l3cache-miss_latency	int	0x0	L3 Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
l3cache-read_access_latency	int	0x0	L3 Cache timing annotation latency for read accesses given in ticks per access (of size l3cache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l3cache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
l3cache-read_latency	int	0x0	L3 Cache timing annotation latency for read accesses given in ticks per byte accessed.l3cache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
l3cache-size	int	0x80000	L3 Cache size in bytes.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-152 Parameters for Cluster_ARM_Cortex-A75 (continued)

Name	Type	Default value	Description
l3cache-snoop_data_transfer_latency	int	0x0	L3 Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
l3cache-snoop_issue_latency	int	0x0	L3 Cache timing annotation latency for snoop accesses issued by this cache in total ticks. This is only used when dcache-state_modelled=true.
l3cache-write_access_latency	int	0x0	L3 Cache timing annotation latency for write accesses given in ticks per access (of size l3cache-write_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l3cache-write_latency is set. This is only used when dcache-state_modelled=true.
l3cache-write_latency	int	0x0	L3 Cache timing annotation latency for write accesses given in ticks per byte accessed. l3cache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
pchannel_treat_simreset_as_poreset	bool	0x0	Register core as ON state to cluster with simulation reset.
periph_address_end	int	0x0	End address for peripheral port address range exclusive(corresponds to AENDMP input signal).
periph_address_start	int	0x0	Start address for peripheral port address range inclusive(corresponds to ASTARTMP input signal).
ptw_latency	int	0x0	Page table walker latency for TA (Timing Annotation), expressed in simulation ticks
tlb_latency	int	0x0	TLB latency for TA (Timing Annotation), expressed in simulation ticks
walk_cache_latency	int	0x0	Walk cache latency for TA (Timing Annotation), expressed in simulation ticks

3.4.26 ARMCortexA76AECT

ARMCortexA76AECT CPU component. This model is written in C++ and models version r0p0 of the RTL.

ARMCortexA76AECT contains the following CADI targets:

- ARM_Cortex-A76AE
- Cluster_ARM_Cortex-A76AE
- PVCache
- TlbCadi

ARMCortexA76AECT contains the following MTI components:

- *4.22 ARM_Cortex-A76AE - trace* on page 4-1556
- *4.48 ARMv8Cluster - trace* on page 4-2401
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402
- *4.68 DSU - trace* on page 4-2463

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

- *4.73 GICv3CPUInterfaceDecoder - trace* on page 4-2489
- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.108 PVCache - trace* on page 4-2761
- *4.114 TLB - trace* on page 4-2834

About ARMCortexA76AECT

The model supports the following features:

- DynamIQ (FCM) system registers.
- Per-core L2 cache.
- A P-Channel for the cluster and for each core.
- Optional peripheral port.
- L3 cache partition.
- Per-core clock.
- `BROADCASTPERSIST` pin.

The model does not support the following features:

- `BROADCASTCACHEMAINTPOU` pin.
- `COREINSTRRET`, `COREINSTRRUN`, or `nPMBIRQ` signals.
- DynamIQ features that are irrelevant to the programmers' view simulation, for example:
  - Automatic CPU retention mode.
  - Level-3 Cache RAM retention.
- 256-bit wide output transactions.
- ECC and parity schemes are hardware-specific so are not supported.
- Self-test features (MBIST).
- Latency configuration.
- Snoop filtering.
- Debug power management signals `DBGPWRDUP` and `DBGRSTREQ` are not implemented, but `DBGPWRUPREQ` and `DBGNOPWRDWN` are implemented.
- Cache stashing capability.
- Split/Lock

This model has a variable number of cores per cluster, specified using the `NUM_CORES` parameter. The following models also exist, with a fixed number of cores per cluster:

- ARMCortexA76AEx2CT.
- ARMCortexA76AEx4CT.
- ARMCortexA76AEx6CT.
- ARMCortexA76AEx8CT.

The per-core parameters are preceded by `cpu`*n*`.`, where *n* identifies the core (0-7).

Table 3-153 Ports

Name	Protocol	Type	Description
`AENDMP`	*Value_64* on page 2-85	Slave	Port to obtain end address of valid peripheral address range (exclusive).
`ASTARTMP`	*Value_64* on page 2-85	Slave	Port to obtain start address of valid peripheral address range (inclusive).
`CNTHPIRQ[8]`	*Signal* on page 2-85	Master	Timer signals to SOC.
`CNTHVIRQ[8]`	*Signal* on page 2-85	Master	Timer signals to SOC.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-153 Ports (continued)

Name	Protocol	Type	Description
CNTPNSIRQ[8]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTPSIRQ[8]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTVIRQ[8]	*Signal* on page 2-85	Master	Timer signals to SOC.
acp_s	PVBus	Slave	AXI ACP slave port.
broadcastatomic	*Signal* on page 2-85	Slave	CHI defined pins.
broadcastcachemaint	*Signal* on page 2-85	Slave	ACE defined pins.
broadcastouter	*Signal* on page 2-85	Slave	ACE defined pins.
broadcastpersist	*Signal* on page 2-85	Slave	CHI defined pins.
cfgend[8]	*Signal* on page 2-85	Slave	This signal if for EE bit initialisation.
cfgte[8]	*Signal* on page 2-85	Slave	This signal provides default exception handling state.
clk_in	ClockSignal	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the cluster level components run e.g cluster level timers, caches and pmu.
clusterid	*Value* on page 2-85	Slave	The port reads the value in CPU ID register field, bits[11:8] of the MPIDR.
clusterpmuirq	*Signal* on page 2-85	Master	DynamIQ pmu irq
cntvalueb	*CounterInterface* on page 2-84	Slave	Interface to SoC level counter module.
commirq[8]	*Signal* on page 2-85	Master	Interrupt signal from debug communications channel.
core_clk_in[8]	ClockSignal	Slave	The clock signal connected to the core_clk_in port is used to determine the rate at which each core executes instructions.
cpuporeset[8]	*Signal* on page 2-85	Slave	Power on reset. Initializes all the processor logic, including debug logic.
cryptodisable[8]	*Signal* on page 2-85	Slave	Disable cryptography extensions after reset.
cti[8]	*v8EmbeddedCrossTrigger_controlprotocol* on page 2-84	Master	Cross trigger matrix port.
ctidbgirq[8]	*Signal* on page 2-85	Master	Cross Trigger Interface (CTI) interrupt trigger output.
dbgen	*Signal* on page 2-85	Slave	External debug interface.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-153 Ports (continued)

Name	Protocol	Type	Description
dbgnopwrdwn[8]	*Signal* on page 2-85	Master	These signals relate to core power down.
dbgpwrupreq[8]	*Signal* on page 2-85	Master	Debug power up request.
dev_debug_s	PVBus	Slave	External debug interface.
event	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE.
fiq[8]	*Signal* on page 2-85	Slave	This signal drives the CPUs fast-interrupt handling.
gicv3_redistributor_s[8]	*GICv3Comms* on page 2-84	Slave	GICv3 AXI-stream port.
irq[8]	*Signal* on page 2-85	Slave	This signal drives the CPUs interrupt handling.
memorymapped_debug_s	PVBus	Slave	External debug interface.
niden	*Signal* on page 2-85	Slave	External debug interface.
pchannel_cluster	*PChannel* on page 2-80	Slave	PChannel for cluster.
pchannel_core[8]	*PChannel* on page 2-80	Slave	PChannels for cores
pmuirq[8]	*Signal* on page 2-85	Master	Interrupt signal from performance monitoring unit.
presetdbg	*Signal* on page 2-85	Slave	Initialize the shared debug APB, Cross Trigger Interface (CTI), and Cross Trigger Matrix (CTM) logic.
pvbus_m0	PVBus	Master	The core will generate bus requests on this port.
pvbus_periph_m	PVBus	Master	The core can generate peripheral bus request on this port.
rei[8]	*Signal* on page 2-85	Slave	Per core RAM Error Interrupt.
reset[8]	*Signal* on page 2-85	Slave	Raising this signal will put the core into reset mode.
rvbaraddr[8]	*Value_64* on page 2-85	Slave	Reset vector base address.
sci_m	SystemCoherencyInterface	Master	System coherency interface port, which is used to take the whole cluster into/out-of coherency domain
sei[8]	*Signal* on page 2-85	Slave	Per core virtual System Error physical pins.
spiden	*Signal* on page 2-85	Slave	External debug interface.
spniden	*Signal* on page 2-85	Slave	External debug interface.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-153 Ports (continued)

Name	Protocol	Type	Description
sporeset	*Signal* on page 2-85	Slave	A single cluster-wide power on reset signal for all resettable registers in DynamIQ.
ticks[8]	*InstructionCount* on page 2-84	Master	This port should be connected to one of the two ticks ports on a 'visualisation' component, in order to display a running instruction count.
vcpumntirq[8]	*Signal* on page 2-85	Master	Interrupt signal for virtual CPU maintenance IRQ.
vfiq[8]	*Signal* on page 2-85	Slave	Virtualised FIQ.
vinithi[8]	*Signal* on page 2-85	Slave	This signal controls of the location of the exception vectors at reset.
virq[8]	*Signal* on page 2-85	Slave	Virtualised IRQ.
virtio_s	PVBus	Slave	The virtio coherent port, hooks directly into the L2 system and becomes coherent (assuming attributes are set correctly).
vsei[8]	*Signal* on page 2-85	Slave	Per core virtual System Error physical pins.

Table 3-154 Parameters for ARM_Cortex-A76AE

Name	Type	Default value	Description
cpu0.CFGEND	bool	0x0	Endianness configuration at reset. 0, little endian. 1, big endian.
cpu0.CFGTE	bool	0x0	Instruction set state when resetting into AArch32. 0, A32. 1, T32.
cpu0.CRYPTODISABLE	bool	0x0	Disable cryptographic features.
cpu0.RVBARADDR	int	0x0	Value of RVBAR_ELx register.
cpu0.VINITHI	bool	0x0	Reset value of SCTLR.V.
cpu0.enable_trace_special_hlt_imm16	bool	0x0	Enable usage of parameter trace_special_hlt_imm16
cpu0.l2cache-hit_latency	int	0x0	L2 Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
cpu0.l2cache-maintenance_latency	int	0x0	L2 Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-154 Parameters for ARM_Cortex-A76AE (continued)

Name	Type	Default value	Description
cpu0.l2cache-miss_latency	int	0x0	L2 Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
cpu0.l2cache-read_access_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
cpu0.l2cache-read_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per byte accessed.l2cache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
cpu0.l2cache-size	int	0x80000	L2 Cache size in bytes.
cpu0.l2cache-snoop_data_transfer_latency	int	0x0	L2 Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
cpu0.l2cache-snoop_issue_latency	int	0x0	L2 Cache timing annotation latency for snoop accesses issued by this cache in total ticks. This is only used when dcache-state_modelled=true.
cpu0.l2cache-write_access_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-write_latency is set. This is only used when dcache-state_modelled=true.
cpu0.l2cache-write_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per byte accessed. l2cache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
cpu0.max_code_cache_mb	int	0x100	Maximum size of the simulation code cache (MiB). For platforms with more than 2 cores this limit will be scaled down. (e.g 1/8 for 16 or more cores)
cpu0.min_sync_level	int	0x0	Force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)
cpu0.semihosting-A32_HLT	int	0xf000	A32 HLT number for semihosting calls.
cpu0.semihosting-A64_HLT	int	0xf000	A64 HLT number for semihosting calls.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-154 Parameters for ARM_Cortex-A76AE (continued)

Name	Type	Default value	Description
cpu0.semihosting-ARM_SVC	int	0x123456	A32 SVC number for semihosting calls.
cpu0.semihosting-T32_HLT	int	0x3c	T32 HLT number for semihosting calls.
cpu0.semihosting-Thumb_SVC	int	0xab	T32 SVC number for semihosting calls.
cpu0.semihosting-cmd_line	string	""	Command line available to semihosting calls.
cpu0.semihosting-cwd	string	""	Base directory for semihosting file access.
cpu0.semihosting-enable	bool	0x1	Enable semihosting SVC/HLT traps.
cpu0.semihosting-heap_base	int	0x0	Virtual address of heap base.
cpu0.semihosting-heap_limit	int	0xf000000	Virtual address of top of heap.
cpu0.semihosting-stack_base	int	0x10000000	Virtual address of base of descending stack.
cpu0.semihosting-stack_limit	int	0xf000000	Virtual address of stack limit.
cpu0.trace_special_hlt_imm16	int	0xf000	For this HLT number, IF enable_trace_special_hlt_imm16=true, skip performing usual HLT execution but call MTI trace if registered
cpu0.vfp-enable_at_reset	bool	0x0	Enable VFP in CPACR, CPPWR, NSACR at reset. Warning: Arm recommends going through the implementation's suggested VFP power-up sequence!

Table 3-155 Parameters for Cluster_ARM_Cortex-A76AE

Name	Type	Default value	Description
BROADCASTATOMIC	bool	0x1	Enable broadcasting of atomic operation. The broadcastatomic signal will override this value if used
BROADCASTCACHEMAINT	bool	0x1	Enable broadcasting of cache maintenance operations to downstream caches. The broadcastcachemaint signal will override this value if used.
BROADCASTOUTER	bool	0x1	Enable broadcasting of Outer Shareable transactions. The broadcastouter signal will override this value if used.
BROADCASTPERSIST	bool	0x1	Enable broadcasting of cache clean to the point of persistence operations. The broadcastpersist signal will override this value if used
CLUSTER_ID	int	0x0	Processor cluster ID value
GICDISABLE	bool	0x1	Disable the new style GICv3 CPU interface in each core model. Should be left enabled unless the platform contains a GICv3 distributor.
cpi_div	int	0x1	Divider for calculating CPI (Cycles Per Instruction)
cpi_mul	int	0x1	Multiplier for calculating CPI (Cycles Per Instruction)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-155 Parameters for Cluster_ARM_Cortex-A76AE (continued)

Name	Type	Default value	Description
dcache-hit_latency	int	0x0	L1 D-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
dcache-maintenance_latency	int	0x0	L1 D-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
dcache-miss_latency	int	0x0	L1 D-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
dcache-prefetch_enabled	bool	0x0	Enable simulation of data cache prefetching. This is only used when dcache-state_modelled=true
dcache-read_access_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per access (of size dcache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
dcache-read_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per byte accessed.dcache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
dcache-snoop_data_transfer_latency	int	0x0	L1 D-Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
dcache-state_modelled	bool	0x0	Set whether D-cache has stateful implementation
dcache-write_access_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per access (of size dcache-write_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-write_latency is set. This is only used when dcache-state_modelled=true.
dcache-write_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per byte accessed. dcache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-155 Parameters for Cluster_ARM_Cortex-A76AE (continued)

Name	Type	Default value	Description
default_opmode	int	0x4	Operating mode of DynamIQ coming out of reset. 0: SFONLY ON, 1: 1/4 CACHE ON, 2: 1/2 CACHE ON, 3: 3/4 CACHE ON, 4: FULL CACHE ON
diagnostics	bool	0x0	Enable DynamIQ diagnostic messages
enable_simulation_performance_optimizations	bool	0x1	With this option enabled, the model will run more quickly, but be less accurate to exact CPU behavior. The model will still be functionally accurate for software, but may increase differences seen between hardware behavior and model behavior for certain workloads (it changes the micro-architectural value of stage12_tlb_size parameter to 1024).
ext_abort_device_read_is_sync	bool	0x0	Synchronous reporting of device-nGnRE read external aborts
ext_abort_device_write_is_sync	bool	0x0	Synchronous reporting of device-nGnRE write external aborts
ext_abort_so_read_is_sync	bool	0x0	Synchronous reporting of device-nGnRnE read external aborts
ext_abort_so_write_is_sync	bool	0x0	Synchronous reporting of device-nGnRnE write external aborts
has_dot_product	int	0x2	Possible values of this parameter are: - 1, feature not implemented. - 2, feature is implemented.
has_peripheral_port	bool	0x0	If true, additional AXI peripheral port is configured.
has_statistical_profiling	bool	0x1	Whether Statistical Based Profiling is implemented
icache-hit_latency	int	0x0	L1 I-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when icache-state_modelled=true.
icache-maintenance_latency	int	0x0	L1 I-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when icache-state_modelled=true.
icache-miss_latency	int	0x0	L1 I-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when icache-state_modelled=true.
icache-prefetch_enabled	bool	0x0	Enable simulation of instruction cache prefetching. This is only used when icache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-155 Parameters for Cluster_ARM_Cortex-A76AE (continued)

Name	Type	Default value	Description
icache-read_access_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per access (of size icache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if icache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when icache-state_modelled=true.
icache-read_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per byte accessed.icache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when icache-state_modelled=true.
icache-state_modelled	bool	0x0	Set whether I-cache has stateful implementation
l3cache-hit_latency	int	0x0	L3 Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
l3cache-maintenance_latency	int	0x0	L3 Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
l3cache-miss_latency	int	0x0	L3 Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
l3cache-read_access_latency	int	0x0	L3 Cache timing annotation latency for read accesses given in ticks per access (of size l3cache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l3cache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
l3cache-read_latency	int	0x0	L3 Cache timing annotation latency for read accesses given in ticks per byte accessed.l3cache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
l3cache-size	int	0x100000	L3 Cache size in bytes.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-155 Parameters for Cluster_ARM_Cortex-A76AE (continued)

Name	Type	Default value	Description
l3cache-snoop_data_transfer_latency	int	0x0	L3 Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
l3cache-snoop_issue_latency	int	0x0	L3 Cache timing annotation latency for snoop accesses issued by this cache in total ticks. This is only used when dcache-state_modelled=true.
l3cache-write_access_latency	int	0x0	L3 Cache timing annotation latency for write accesses given in ticks per access (of size l3cache-write_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l3cache-write_latency is set. This is only used when dcache-state_modelled=true.
l3cache-write_latency	int	0x0	L3 Cache timing annotation latency for write accesses given in ticks per byte accessed. l3cache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
pchannel_treat_simreset_as_poreset	bool	0x0	Register core as ON state to cluster with simulation reset.
periph_address_end	int	0x0	End address for peripheral port address range exclusive(corresponds to AENDMP input signal).
periph_address_start	int	0x0	Start address for peripheral port address range inclusive(corresponds to ASTARTMP input signal).
ptw_latency	int	0x0	Page table walker latency for TA (Timing Annotation), expressed in simulation ticks
tlb_latency	int	0x0	TLB latency for TA (Timing Annotation), expressed in simulation ticks
treat-dcache-cmos-to-pou-as-nop	bool	0x0	Whether dcache invalidation to the point of unification is required for instruction to data coherence. true - Invalidate operations not required
walk_cache_latency	int	0x0	Walk cache latency for TA (Timing Annotation), expressed in simulation ticks

3.4.27 ARMCortexA76CT

ARMCortexA76CT CPU component. This model is written in C++ and models version r0p0 of the RTL.

ARMCortexA76CT contains the following CADI targets:

- ARM_Cortex-A76
- Cluster_ARM_Cortex-A76
- PVCache
- TlbCadi

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ARMCortexA76CT contains the following MTI components:

- *4.21 ARM_Cortex-A76 - trace* on page 4-1517
- *4.48 ARMv8Cluster - trace* on page 4-2401
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402
- *4.68 DSU - trace* on page 4-2463
- *4.73 GICv3CPUInterfaceDecoder - trace* on page 4-2489
- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.108 PVCache - trace* on page 4-2761
- *4.114 TLB - trace* on page 4-2834

About ARMCortexA76CT

The model supports the following features:

- DynamIQ (FCM) system registers.
- Per-core L2 cache.
- A P-Channel for the cluster and for each core.
- Optional peripheral port.
- L3 cache partition.
- Per-core clock.
- `BROADCASTPERSIST` pin.

The model does not support the following features:

- `BROADCASTCACHEMAINTPOU` pin.
- `COREINSTRRET`, `COREINSTRRUN`, or `nPMBIRQ` signals.
- DynamIQ features that are irrelevant to the programmers' view simulation, for example:
  - Automatic CPU retention mode.
  - Level-3 Cache RAM retention.
- 256-bit wide output transactions.
- ECC and parity schemes are hardware-specific so are not supported.
- Self-test features (MBIST).
- Latency configuration.
- Snoop filtering.
- Debug power management signals `DBGPWRDUP` and `DBGRSTREQ` are not implemented, but `DBGPWRUPREQ` and `DBGNOPWRDWN` are implemented.
- Cache stashing capability.

This component has a variable number of cores per cluster, specified using the `NUM_CORES` parameter. The following components also exist, with a fixed number of cores per cluster:

- ARMCortexA76x1CT.
- ARMCortexA76x2CT.
- ARMCortexA76x3CT.
- ARMCortexA76x4CT.

The per-core parameters are preceded by `cpu`*n*`.`, where *n* identifies the core (0-3).

Table 3-156 Ports

Name	Protocol	Type	Description
`AENDMP`	*`Value_64` on page 2-85*	Slave	Port to obtain end address of valid peripheral address range (exclusive).
`ASTARTMP`	*`Value_64` on page 2-85*	Slave	Port to obtain start address of valid peripheral address range (inclusive).

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-156 Ports (continued)

Name	Protocol	Type	Description
CNTHPIRQ[4]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTHVIRQ[4]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTPNSIRQ[4]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTPSIRQ[4]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTVIRQ[4]	*Signal* on page 2-85	Master	Timer signals to SOC.
acp_s	PVBus	Slave	AXI ACP slave port.
broadcastatomic	*Signal* on page 2-85	Slave	CHI defined pins.
broadcastcachemaint	*Signal* on page 2-85	Slave	ACE defined pins.
broadcastouter	*Signal* on page 2-85	Slave	ACE defined pins.
broadcastpersist	*Signal* on page 2-85	Slave	CHI defined pins.
cfgend[4]	*Signal* on page 2-85	Slave	This signal if for EE bit initialisation.
cfgte[4]	*Signal* on page 2-85	Slave	This signal provides default exception handling state.
clk_in	ClockSignal	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the cluster level components run e.g cluster level timers, caches and pmu.
clusterid	*Value* on page 2-85	Slave	The port reads the value in CPU ID register field, bits[11:8] of the MPIDR.
clusterpmuirq	*Signal* on page 2-85	Master	DynamIQ pmu irq
cntvalueb	*CounterInterface* on page 2-84	Slave	Interface to SoC level counter module.
commirq[4]	*Signal* on page 2-85	Master	Interrupt signal from debug communications channel.
core_clk_in[4]	ClockSignal	Slave	The clock signal connected to the core_clk_in port is used to determine the rate at which each core executes instructions.
cp15sdisable[4]	*Signal* on page 2-85	Slave	This signal disables write access to some system control processor registers.
cpuporeset[4]	*Signal* on page 2-85	Slave	Power on reset. Initializes all the processor logic, including debug logic.
cryptodisable[4]	*Signal* on page 2-85	Slave	Disable cryptography extensions after reset.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-156 Ports (continued)

Name	Protocol	Type	Description
cti[4]	*v8EmbeddedCrossTrigger_controlprotocol* on page 2-84	Master	Cross trigger matrix port.
ctidbgirq[4]	*Signal* on page 2-85	Master	Cross Trigger Interface (CTI) interrupt trigger output.
dbgen[4]	*Signal* on page 2-85	Slave	External debug interface.
dbgnopwrdwn[4]	*Signal* on page 2-85	Master	These signals relate to core power down.
dbgpwrupreq[4]	*Signal* on page 2-85	Master	Debug power up request.
dev_debug_s	PVBus	Slave	External debug interface.
event	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE.
fiq[4]	*Signal* on page 2-85	Slave	This signal drives the CPUs fast-interrupt handling.
gicv3_redistributor_s[4]	*GICv3Comms* on page 2-84	Slave	GICv3 AXI-stream port.
irq[4]	*Signal* on page 2-85	Slave	This signal drives the CPUs interrupt handling.
memorymapped_debug_s	PVBus	Slave	External debug interface.
niden[4]	*Signal* on page 2-85	Slave	External debug interface.
pchannel_cluster	*PChannel* on page 2-80	Slave	PChannel for cluster.
pchannel_core[4]	*PChannel* on page 2-80	Slave	PChannels for cores
pmuirq[4]	*Signal* on page 2-85	Master	Interrupt signal from performance monitoring unit.
presetdbg	*Signal* on page 2-85	Slave	Initialize the shared debug APB, Cross Trigger Interface (CTI), and Cross Trigger Matrix (CTM) logic.
pvbus_m0	PVBus	Master	The core will generate bus requests on this port.
pvbus_periph_m	PVBus	Master	The core can generate peripheral bus request on this port.
rei[4]	*Signal* on page 2-85	Slave	Per core RAM Error Interrupt.
reset[4]	*Signal* on page 2-85	Slave	Raising this signal will put the core into reset mode.
rvbaraddr[4]	*Value_64* on page 2-85	Slave	Reset vector base address.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-156 Ports (continued)

Name	Protocol	Type	Description
sci_m	SystemCoherencyInterface	Master	System coherency interface port, which is used to take the whole cluster into/out-of coherency domain
sei[4]	*Signal* on page 2-85	Slave	Per core virtual System Error physical pins.
spiden[4]	*Signal* on page 2-85	Slave	External debug interface.
spniden[4]	*Signal* on page 2-85	Slave	External debug interface.
sporeset	*Signal* on page 2-85	Slave	A single cluster-wide power on reset signal for all resettable registers in DynamIQ.
ticks[4]	*InstructionCount* on page 2-84	Master	This port should be connected to one of the two ticks ports on a 'visualisation' component, in order to display a running instruction count.
vcpumntirq[4]	*Signal* on page 2-85	Master	Interrupt signal for virtual CPU maintenance IRQ.
vfiq[4]	*Signal* on page 2-85	Slave	Virtualised FIQ.
vinithi[4]	*Signal* on page 2-85	Slave	This signal controls of the location of the exception vectors at reset.
virq[4]	*Signal* on page 2-85	Slave	Virtualised IRQ.
virtio_s	PVBus	Slave	The virtio coherent port, hooks directly into the L2 system and becomes coherent (assuming attributes are set correctly).
vsei[4]	*Signal* on page 2-85	Slave	Per core virtual System Error physical pins.

Table 3-157 Parameters for Cluster_ARM_Cortex-A76

Name	Type	Default value	Description
BROADCASTATOMIC	bool	0x1	Enable broadcasting of atomic operation. The broadcastatomic signal will override this value if used
BROADCASTCACHEMAINT	bool	0x1	Enable broadcasting of cache maintenance operations to downstream caches. The broadcastcachemaint signal will override this value if used.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-157 Parameters for Cluster_ARM_Cortex-A76 (continued)

Name	Type	Default value	Description
BROADCASTOUTER	bool	0x1	Enable broadcasting of Outer Shareable transactions. The broadcastouter signal will override this value if used.
BROADCASTPERSIST	bool	0x1	Enable broadcasting of cache clean to the point of persistence operations. The broadcastpersist signal will override this value if used
CLUSTER_ID	int	0x0	Processor cluster ID value
GICDISABLE	bool	0x1	Disable the new style GICv3 CPU interface in each core model. Should be left enabled unless the platform contains a GICv3 distributor.
cpi_div	int	0x1	Divider for calculating CPI (Cycles Per Instruction)
cpi_mul	int	0x1	Multiplier for calculating CPI (Cycles Per Instruction)
dcache-hit_latency	int	0x0	L1 D-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
dcache-maintenance_latency	int	0x0	L1 D-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
dcache-miss_latency	int	0x0	L1 D-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
dcache-prefetch_enabled	bool	0x0	Enable simulation of data cache prefetching. This is only used when dcache-state_modelled=true
dcache-read_access_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per access (of size dcache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
dcache-read_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per byte accessed.dcache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-157 Parameters for Cluster_ARM_Cortex-A76 (continued)

Name	Type	Default value	Description
dcache-snoop_data_transfer_latency	int	0x0	L1 D-Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
dcache-state_modelled	bool	0x0	Set whether D-cache has stateful implementation
dcache-write_access_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per access (of size dcache-write_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-write_latency is set. This is only used when dcache-state_modelled=true.
dcache-write_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per byte accessed. dcache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
default_opmode	int	0x4	Operating mode of DynamIQ coming out of reset. 0: SFONLY ON, 1: 1/4 CACHE ON, 2: 1/2 CACHE ON, 3: 3/4 CACHE ON, 4: FULL CACHE ON
diagnostics	bool	0x0	Enable DynamIQ diagnostic messages
enable_simulation_performance_optimizations	bool	0x1	With this option enabled, the model will run more quickly, but be less accurate to exact CPU behavior. The model will still be functionally accurate for software, but may increase differences seen between hardware behavior and model behavior for certain workloads (it changes the micro-architectural value of stage12_tlb_size parameter to 1024).
ext_abort_device_read_is_sync	bool	0x0	Synchronous reporting of device-nGnRE read external aborts
ext_abort_device_write_is_sync	bool	0x0	Synchronous reporting of device-nGnRE write external aborts
ext_abort_so_read_is_sync	bool	0x0	Synchronous reporting of device-nGnRnE read external aborts
ext_abort_so_write_is_sync	bool	0x0	Synchronous reporting of device-nGnRnE write external aborts
has_peripheral_port	bool	0x0	If true, additional AXI peripheral port is configured.
has_statistical_profiling	bool	0x1	Whether Statistical Based Profiling is implemented
icache-hit_latency	int	0x0	L1 I-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when icache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-157 Parameters for Cluster_ARM_Cortex-A76 (continued)

Name	Type	Default value	Description
icache-maintenance_latency	int	0x0	L1 I-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when icache-state_modelled=true.
icache-miss_latency	int	0x0	L1 I-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when icache-state_modelled=true.
icache-prefetch_enabled	bool	0x0	Enable simulation of instruction cache prefetching. This is only used when icache-state_modelled=true.
icache-read_access_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per access (of size icache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if icache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when icache-state_modelled=true.
icache-read_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per byte accessed.icache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when icache-state_modelled=true.
icache-state_modelled	bool	0x0	Set whether I-cache has stateful implementation
l3cache-hit_latency	int	0x0	L3 Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
l3cache-maintenance_latency	int	0x0	L3 Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
l3cache-miss_latency	int	0x0	L3 Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
l3cache-read_access_latency	int	0x0	L3 Cache timing annotation latency for read accesses given in ticks per access (of size l3cache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l3cache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-157 Parameters for Cluster_ARM_Cortex-A76 (continued)

Name	Type	Default value	Description
l3cache-read_latency	int	0x0	L3 Cache timing annotation latency for read accesses given in ticks per byte accessed.l3cache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
l3cache-size	int	0x100000	L3 Cache size in bytes.
l3cache-snoop_data_transfer_latency	int	0x0	L3 Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
l3cache-snoop_issue_latency	int	0x0	L3 Cache timing annotation latency for snoop accesses issued by this cache in total ticks. This is only used when dcache-state_modelled=true.
l3cache-write_access_latency	int	0x0	L3 Cache timing annotation latency for write accesses given in ticks per access (of size l3cache-write_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l3cache-write_latency is set. This is only used when dcache-state_modelled=true.
l3cache-write_latency	int	0x0	L3 Cache timing annotation latency for write accesses given in ticks per byte accessed. l3cache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
pchannel_treat_simreset_as_poreset	bool	0x0	Register core as ON state to cluster with simulation reset.
periph_address_end	int	0x0	End address for peripheral port address range exclusive(corresponds to AENDMP input signal).
periph_address_start	int	0x0	Start address for peripheral port address range inclusive(corresponds to ASTARTMP input signal).
ptw_latency	int	0x0	Page table walker latency for TA (Timing Annotation), expressed in simulation ticks
tlb_latency	int	0x0	TLB latency for TA (Timing Annotation), expressed in simulation ticks
treat-dcache-cmos-to-pou-as-nop	bool	0x0	Whether dcache invalidation to the point of unification is required for instruction to data coherence. true - Invalidate operations not required
walk_cache_latency	int	0x0	Walk cache latency for TA (Timing Annotation), expressed in simulation ticks

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-158 Parameters for ARM_Cortex-A76

Name	Type	Default value	Description
cpu0.CFGEND	bool	0x0	Endianness configuration at reset. 0, little endian. 1, big endian.
cpu0.CFGTE	bool	0x0	Instruction set state when resetting into AArch32. 0, A32. 1, T32.
cpu0.CRYPTODISABLE	bool	0x0	Disable cryptographic features.
cpu0.RVBARADDR	int	0x0	Value of RVBAR_ELx register.
cpu0.VINITHI	bool	0x0	Reset value of SCTLR.V.
cpu0.enable_trace_special_hlt_imm16	bool	0x0	Enable usage of parameter trace_special_hlt_imm16
cpu0.l2cache-hit_latency	int	0x0	L2 Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
cpu0.l2cache-maintenance_latency	int	0x0	L2 Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
cpu0.l2cache-miss_latency	int	0x0	L2 Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
cpu0.l2cache-read_access_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
cpu0.l2cache-read_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per byte accessed.l2cache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
cpu0.l2cache-size	int	0x80000	L2 Cache size in bytes.
cpu0.l2cache-snoop_data_transfer_latency	int	0x0	L2 Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
cpu0.l2cache-snoop_issue_latency	int	0x0	L2 Cache timing annotation latency for snoop accesses issued by this cache in total ticks. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-158 Parameters for ARM_Cortex-A76 (continued)

Name	Type	Default value	Description
cpu0.l2cache-write_access_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-write_latency is set. This is only used when dcache-state_modelled=true.
cpu0.l2cache-write_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per byte accessed. l2cache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
cpu0.max_code_cache_mb	int	0x100	Maximum size of the simulation code cache (MiB). For platforms with more than 2 cores this limit will be scaled down. (e.g 1/8 for 16 or more cores)
cpu0.min_sync_level	int	0x0	Force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)
cpu0.semihosting-A32_HLT	int	0xf000	A32 HLT number for semihosting calls.
cpu0.semihosting-A64_HLT	int	0xf000	A64 HLT number for semihosting calls.
cpu0.semihosting-ARM_SVC	int	0x123456	A32 SVC number for semihosting calls.
cpu0.semihosting-T32_HLT	int	0x3c	T32 HLT number for semihosting calls.
cpu0.semihosting-Thumb_SVC	int	0xab	T32 SVC number for semihosting calls.
cpu0.semihosting-cmd_line	string	""	Command line available to semihosting calls.
cpu0.semihosting-cwd	string	""	Base directory for semihosting file access.
cpu0.semihosting-enable	bool	0x1	Enable semihosting SVC/HLT traps.
cpu0.semihosting-heap_base	int	0x0	Virtual address of heap base.
cpu0.semihosting-heap_limit	int	0xf000000	Virtual address of top of heap.
cpu0.semihosting-stack_base	int	0x10000000	Virtual address of base of descending stack.
cpu0.semihosting-stack_limit	int	0xf000000	Virtual address of stack limit.
cpu0.trace_special_hlt_imm16	int	0xf000	For this HLT number, IF enable_trace_special_hlt_imm16=true, skip performing usual HLT execution but call MTI trace if registered
cpu0.vfp-enable_at_reset	bool	0x0	Enable VFP in CPACR, CPPWR, NSACR at reset. Warning: Arm recommends going through the implementation's suggested VFP power-up sequence!

3.4.28 ARMCortexA77CT

ARMCortexA77CT CPU component. This model is written in C++ and models version r0p0 of the RTL.

ARMCortexA77CT contains the following CADI targets:

- ARM_Cortex-A77
- Cluster_ARM_Cortex-A77

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

- PVCache
- TlbCadi

ARMCortexA77CT contains the following MTI components:
- *4.23 ARM_Cortex-A77 - trace* on page 4-1595
- *4.48 ARMv8Cluster - trace* on page 4-2401
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402
- *4.68 DSU - trace* on page 4-2463
- *4.73 GICv3CPUInterfaceDecoder - trace* on page 4-2489
- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.108 PVCache - trace* on page 4-2761
- *4.114 TLB - trace* on page 4-2834

About ARMCortexA77CT

The model supports the following features:

- DynamIQ (FCM) system registers.
- Per-core L2 cache.
- A P-Channel for the cluster and for each core.
- Optional peripheral port.
- L3 cache partition.
- Per-core clock.
- `BROADCASTPERSIST` pin.

The model does not support the following features:

- `BROADCASTCACHEMAINTPOU` pin.
- `COREINSTRRET`, `COREINSTRRUN`, or `nPMBIRQ` signals.
- DynamIQ features that are irrelevant to the programmers' view simulation, for example:
  - Automatic CPU retention mode.
  - Level-3 Cache RAM retention.
- 256-bit wide output transactions.
- ECC and parity schemes are hardware-specific so are not supported.
- Self-test features (MBIST).
- Latency configuration.
- Snoop filtering.
- Debug power management signals `DBGPWRDUP` and `DBGRSTREQ` are not implemented, but `DBGPWRUPREQ` and `DBGNOPWRDWN` are implemented.
- Cache stashing capability.

This model has a variable number of cores per cluster, specified using the `NUM_CORES` parameter. The following models also exist, with a fixed number of cores per cluster:
- ARMCortexA77x1CT.
- ARMCortexA77x2CT.
- ARMCortexA77x3CT.
- ARMCortexA77x4CT.

The per-core parameters are preceded by `cpu`*`n`*`.`, where *n* identifies the core (0-3).

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-159 Ports

Name	Protocol	Type	Description
AENDMP	*Value_64* on page 2-85	Slave	Port to obtain end address of valid peripheral address range (exclusive).
ASTARTMP	*Value_64* on page 2-85	Slave	Port to obtain start address of valid peripheral address range (inclusive).
CNTHPIRQ[4]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTHVIRQ[4]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTPNSIRQ[4]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTPSIRQ[4]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTVIRQ[4]	*Signal* on page 2-85	Master	Timer signals to SOC.
acp_s	PVBus	Slave	AXI ACP slave port.
broadcastatomic	*Signal* on page 2-85	Slave	CHI defined pins.
broadcastcachemaint	*Signal* on page 2-85	Slave	ACE defined pins.
broadcastouter	*Signal* on page 2-85	Slave	ACE defined pins.
broadcastpersist	*Signal* on page 2-85	Slave	CHI defined pins.
cfgend[4]	*Signal* on page 2-85	Slave	This signal if for EE bit initialisation.
cfgsdisable	*Signal* on page 2-85	Slave	This signal disables write access to some secure Interrupt Controller registers.
cfgte[4]	*Signal* on page 2-85	Slave	This signal provides default exception handling state.
clk_in	ClockSignal	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the cluster level components run e.g cluster level timers, caches and pmu.
clusterid	*Value* on page 2-85	Slave	The port reads the value in CPU ID register field, bits[11:8] of the MPIDR.
clusterpmuirq	*Signal* on page 2-85	Master	DynamIQ pmu irq
cntvalueb	*CounterInterface* on page 2-84	Slave	Interface to SoC level counter module.
commirq[4]	*Signal* on page 2-85	Master	Interrupt signal from debug communications channel.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-159 Ports (continued)

Name	Protocol	Type	Description
core_clk_in[4]	ClockSignal	Slave	The clock signal connected to the core_clk_in port is used to determine the rate at which each core executes instructions.
cpuporeset[4]	*Signal* on page 2-85	Slave	CPU power on reset. Initializes all the processor logic, including debug logic.
cryptodisable[4]	*Signal* on page 2-85	Slave	Disable cryptography extensions after reset.
cti[4]	*v8EmbeddedCrossTrigger_controlprotocol* on page 2-84	Master	Cross trigger matrix port.
ctidbgirq[4]	*Signal* on page 2-85	Master	Cross Trigger Interface (CTI) interrupt trigger output.
dbgen[4]	*Signal* on page 2-85	Slave	External debug interface.
dbgnopwrdwn[4]	*Signal* on page 2-85	Master	No power-down request.
dbgpwrupreq[4]	*Signal* on page 2-85	Master	Debug power up request.
dev_debug_s	PVBus	Slave	External debug interface.
event	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE.
fiq[4]	*Signal* on page 2-85	Slave	This signal drives the CPUs fast-interrupt handling.
gicv3_redistributor_s[4]	*GICv3Comms* on page 2-84	Slave	GICv3 AXI-stream port.
irq[4]	*Signal* on page 2-85	Slave	This signal drives the CPUs interrupt handling.
memorymapped_debug_s	PVBus	Slave	External debug interface.
niden[4]	*Signal* on page 2-85	Slave	External debug interface.
pchannel_cluster	*PChannel* on page 2-80	Slave	PChannel for cluster.
pchannel_core[4]	*PChannel* on page 2-80	Slave	PChannels for cores
pmuirq[4]	*Signal* on page 2-85	Master	Interrupt signal from performance monitoring unit.
presetdbg	*Signal* on page 2-85	Slave	Initialize the shared debug APB, Cross Trigger Interface (CTI), and Cross Trigger Matrix (CTM) logic.
pvbus_m0	PVBus	Master	The core will generate bus requests on this port.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-159 Ports (continued)

Name	Protocol	Type	Description
pvbus_periph_m	PVBus	Master	The core can generate peripheral bus request on this port.
rei[4]	*Signal* on page 2-85	Slave	Per core RAM Error Interrupt.
reset[4]	*Signal* on page 2-85	Slave	Raising this signal will put the core into reset mode.
rvbaraddr[4]	*Value_64* on page 2-85	Slave	Reset vector base address.
sci_m	SystemCoherencyInterface	Master	System coherency interface port, which is used to take the whole cluster into/out-of coherency domain
sei[4]	*Signal* on page 2-85	Slave	Per core System Error physical pins.
spiden[4]	*Signal* on page 2-85	Slave	External debug interface.
spniden[4]	*Signal* on page 2-85	Slave	External debug interface.
sporeset	*Signal* on page 2-85	Slave	A single cluster-wide power on reset signal for all resettable registers in DynamIQ.
ticks[4]	*InstructionCount* on page 2-84	Master	This port should be connected to one of the two ticks ports on a 'visualisation' component, in order to display a running instruction count.
vcpumntirq[4]	*Signal* on page 2-85	Master	Interrupt signal for virtual CPU maintenance IRQ.
vfiq[4]	*Signal* on page 2-85	Slave	Virtualised FIQ.
vinithi[4]	*Signal* on page 2-85	Slave	This signal controls of the location of the exception vectors at reset.
virq[4]	*Signal* on page 2-85	Slave	Virtualised IRQ.
virtio_s	PVBus	Slave	The virtio coherent port, hooks directly into the L2 system and becomes coherent (assuming attributes are set correctly).
vsei[4]	*Signal* on page 2-85	Slave	Per core virtual System Error physical pins.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-160 Parameters for ARM_Cortex-A77

Name	Type	Default value	Description
cpu0.CFGEND	bool	0x0	Endianness configuration at reset. 0, little endian. 1, big endian.
cpu0.CFGTE	bool	0x0	Instruction set state when resetting into AArch32. 0, A32. 1, T32.
cpu0.CRYPTODISABLE	bool	0x0	Disable cryptographic features.
cpu0.RVBARADDR	int	0x0	Value of RVBAR_ELx register.
cpu0.VINITHI	bool	0x0	Reset value of SCTLR.V.
cpu0.enable_trace_special_hlt_imm16	bool	0x0	Enable usage of parameter trace_special_hlt_imm16
cpu0.l2cache-hit_latency	int	0x0	L2 Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
cpu0.l2cache-maintenance_latency	int	0x0	L2 Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
cpu0.l2cache-miss_latency	int	0x0	L2 Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
cpu0.l2cache-read_access_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
cpu0.l2cache-read_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per byte accessed.l2cache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
cpu0.l2cache-size	int	0x80000	L2 Cache size in bytes.
cpu0.l2cache-snoop_data_transfer_latency	int	0x0	L2 Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
cpu0.l2cache-snoop_issue_latency	int	0x0	L2 Cache timing annotation latency for snoop accesses issued by this cache in total ticks. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-160 Parameters for ARM_Cortex-A77 (continued)

Name	Type	Default value	Description
cpu0.l2cache-write_access_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-write_latency is set. This is only used when dcache-state_modelled=true.
cpu0.l2cache-write_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per byte accessed. l2cache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
cpu0.max_code_cache_mb	int	0x100	Maximum size of the simulation code cache (MiB). For platforms with more than 2 cores this limit will be scaled down. (e.g 1/8 for 16 or more cores)
cpu0.min_sync_level	int	0x0	Force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)
cpu0.semihosting-A32_HLT	int	0xf000	A32 HLT number for semihosting calls.
cpu0.semihosting-A64_HLT	int	0xf000	A64 HLT number for semihosting calls.
cpu0.semihosting-ARM_SVC	int	0x123456	A32 SVC number for semihosting calls.
cpu0.semihosting-T32_HLT	int	0x3c	T32 HLT number for semihosting calls.
cpu0.semihosting-Thumb_SVC	int	0xab	T32 SVC number for semihosting calls.
cpu0.semihosting-cmd_line	string	""	Command line available to semihosting calls.
cpu0.semihosting-cwd	string	""	Base directory for semihosting file access.
cpu0.semihosting-enable	bool	0x1	Enable semihosting SVC/HLT traps.
cpu0.semihosting-heap_base	int	0x0	Virtual address of heap base.
cpu0.semihosting-heap_limit	int	0xf000000	Virtual address of top of heap.
cpu0.semihosting-stack_base	int	0x10000000	Virtual address of base of descending stack.
cpu0.semihosting-stack_limit	int	0xf000000	Virtual address of stack limit.
cpu0.trace_special_hlt_imm16	int	0xf000	For this HLT number, IF enable_trace_special_hlt_imm16=true, skip performing usual HLT execution but call MTI trace if registered
cpu0.vfp-enable_at_reset	bool	0x0	Enable VFP in CPACR, CPPWR, NSACR at reset. Warning: Arm recommends going through the implementation's suggested VFP power-up sequence!

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-161 Parameters for Cluster_ARM_Cortex-A77

Name	Type	Default value	Description
BROADCASTATOMIC	bool	0x1	Enable broadcasting of atomic operation. The broadcastatomic signal will override this value if used
BROADCASTCACHEMAINT	bool	0x1	Enable broadcasting of cache maintenance operations to downstream caches. The broadcastcachemaint signal will override this value if used.
BROADCASTOUTER	bool	0x1	Enable broadcasting of Outer Shareable transactions. The broadcastouter signal will override this value if used.
BROADCASTPERSIST	bool	0x1	Enable broadcasting of cache clean to the point of persistence operations. The broadcastpersist signal will override this value if used
CLUSTER_ID	int	0x0	Processor cluster ID value
GICDISABLE	bool	0x1	Disable the new style GICv3 CPU interface in each core model. Should be left enabled unless the platform contains a GICv3 distributor.
cpi_div	int	0x1	Divider for calculating CPI (Cycles Per Instruction)
cpi_mul	int	0x1	Multiplier for calculating CPI (Cycles Per Instruction)
dcache-hit_latency	int	0x0	L1 D-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
dcache-maintenance_latency	int	0x0	L1 D-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
dcache-miss_latency	int	0x0	L1 D-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
dcache-prefetch_enabled	bool	0x0	Enable simulation of data cache prefetching. This is only used when dcache-state_modelled=true
dcache-read_access_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per access (of size dcache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-161 Parameters for Cluster_ARM_Cortex-A77 (continued)

Name	Type	Default value	Description
dcache-read_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per byte accessed.dcache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
dcache-snoop_data_transfer_latency	int	0x0	L1 D-Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
dcache-state_modelled	bool	0x0	Set whether D-cache has stateful implementation
dcache-write_access_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per access (of size dcache-write_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-write_latency is set. This is only used when dcache-state_modelled=true.
dcache-write_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per byte accessed. dcache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
default_opmode	int	0x4	Operating mode of DynamIQ coming out of reset. 0: SFONLY ON, 1: 1/4 CACHE ON, 2: 1/2 CACHE ON, 3: 3/4 CACHE ON, 4: FULL CACHE ON
diagnostics	bool	0x0	Enable DynamIQ diagnostic messages
enable_simulation_performance_optimizations	bool	0x1	With this option enabled, the model will run more quickly, but be less accurate to exact CPU behavior. The model will still be functionally accurate for software, but may increase differences seen between hardware behavior and model behavior for certain workloads (it changes the micro-architectural value of stage12_tlb_size parameter to 1024).
ext_abort_device_read_is_sync	bool	0x0	Synchronous reporting of device-nGnRE read external aborts
ext_abort_device_write_is_sync	bool	0x0	Synchronous reporting of device-nGnRE write external aborts
ext_abort_so_read_is_sync	bool	0x0	Synchronous reporting of device-nGnRnE read external aborts
ext_abort_so_write_is_sync	bool	0x0	Synchronous reporting of device-nGnRnE write external aborts

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-161 Parameters for Cluster_ARM_Cortex-A77 (continued)

Name	Type	Default value	Description
has_peripheral_port	bool	0x0	If true, additional AXI peripheral port is configured.
has_statistical_profiling	bool	0x1	Whether Statistical Based Profiling is implemented
icache-hit_latency	int	0x0	L1 I-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when icache-state_modelled=true.
icache-maintenance_latency	int	0x0	L1 I-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when icache-state_modelled=true.
icache-miss_latency	int	0x0	L1 I-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when icache-state_modelled=true.
icache-prefetch_enabled	bool	0x0	Enable simulation of instruction cache prefetching. This is only used when icache-state_modelled=true.
icache-read_access_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per access (of size icache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if icache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when icache-state_modelled=true.
icache-read_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per byte accessed.icache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when icache-state_modelled=true.
icache-state_modelled	bool	0x0	Set whether I-cache has stateful implementation
l3cache-hit_latency	int	0x0	L3 Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
l3cache-maintenance_latency	int	0x0	L3 Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
l3cache-miss_latency	int	0x0	L3 Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-161 Parameters for Cluster_ARM_Cortex-A77 (continued)

Name	Type	Default value	Description
l3cache-read_access_latency	int	0x0	L3 Cache timing annotation latency for read accesses given in ticks per access (of size l3cache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l3cache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
l3cache-read_latency	int	0x0	L3 Cache timing annotation latency for read accesses given in ticks per byte accessed.l3cache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
l3cache-size	int	0x100000	L3 Cache size in bytes.
l3cache-snoop_data_transfer_latency	int	0x0	L3 Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
l3cache-snoop_issue_latency	int	0x0	L3 Cache timing annotation latency for snoop accesses issued by this cache in total ticks. This is only used when dcache-state_modelled=true.
l3cache-write_access_latency	int	0x0	L3 Cache timing annotation latency for write accesses given in ticks per access (of size l3cache-write_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l3cache-write_latency is set. This is only used when dcache-state_modelled=true.
l3cache-write_latency	int	0x0	L3 Cache timing annotation latency for write accesses given in ticks per byte accessed. l3cache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
pchannel_treat_simreset_as_poreset	bool	0x0	Register core as ON state to cluster with simulation reset.
periph_address_end	int	0x0	End address for peripheral port address range exclusive(corresponds to AENDMP input signal).
periph_address_start	int	0x0	Start address for peripheral port address range inclusive(corresponds to ASTARTMP input signal).
ptw_latency	int	0x0	Page table walker latency for TA (Timing Annotation), expressed in simulation ticks

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-161 Parameters for Cluster_ARM_Cortex-A77 (continued)

Name	Type	Default value	Description
`tlb_latency`	`int`	`0x0`	TLB latency for TA (Timing Annotation), expressed in simulation ticks
`treat-dcache-cmos-to-pou-as-nop`	`bool`	`0x0`	Whether dcache invalidation to the point of unification is required for instruction to data coherence. true - Invalidate operations not required
`walk_cache_latency`	`int`	`0x0`	Walk cache latency for TA (Timing Annotation), expressed in simulation ticks

3.4.29 ARMCortexA78CCT

ARMCortexA78CCT CPU component. This model is written in C++ and models version r0p0 of the RTL.

ARMCortexA78CCT contains the following CADI targets:

- ARM_Cortex-A78C
- Cluster_ARM_Cortex-A78C
- PVCache
- TlbCadi

ARMCortexA78CCT contains the following MTI components:

- *4.25 ARM_Cortex-A78C - trace* on page 4-1673
- *4.48 ARMv8Cluster - trace* on page 4-2401
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402
- *4.68 DSU - trace* on page 4-2463
- *4.73 GICv3CPUInterfaceDecoder - trace* on page 4-2489
- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.108 PVCache - trace* on page 4-2761
- *4.114 TLB - trace* on page 4-2834

About ARMCortexA78CCT

- DynamIQ (FCM) system registers.
- Per-core L2 cache.
- A P-Channel for the cluster and for each core.
- Optional peripheral port.
- L3 cache partition.
- Per-core clock.
- `BROADCASTPERSIST` pin.

The model does not support the following features:

- `BROADCASTCACHEMAINTPOU` pin.
- `COREINSTRRET`, `COREINSTRRUN`, or `nPMBIRQ` signals.
- DynamIQ features that are irrelevant to the programmers' view simulation, for example:
  - Automatic CPU retention mode.
  - Level-3 Cache RAM retention.
- 256-bit wide output transactions.
- ECC and parity schemes are hardware-specific so are not supported.
- Self-test features (MBIST).
- Latency configuration.
- Snoop filtering.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

- Debug power management signals DBGPWRDUP and DBGRSTREQ are not implemented, but DBGPWRUPREQ and DBGNOPWRDWN are implemented.
- Cache stashing capability.
- Each thread has its own L1Cache and L2Cache instead of a common cache that is shared by all threads of the core.
- Per-thread parameters are not yet supported but signals are implemented.

This component has a variable number of cores per cluster, specified using the NUM_CORES parameter. The following components also exist, with a fixed number of cores per cluster:

- ARMCortexA78x1CCT.
- ARMCortexA78x2CCT.
- ARMCortexA78x4CCT.
- ARMCortexA78x6CCT.
- ARMCortexA78x8CCT.

The per-core parameters are preceded by cpu*n*., where *n* identifies the core (0-7).

Table 3-162 Ports

Name	Protocol	Type	Description
AENDMP	*Value_64* on page 2-85	Slave	Port to obtain end address of valid peripheral address range (exclusive).
ASTARTMP	*Value_64* on page 2-85	Slave	Port to obtain start address of valid peripheral address range (inclusive).
CNTHPIRQ[8]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTHVIRQ[8]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTPNSIRQ[8]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTPSIRQ[8]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTVIRQ[8]	*Signal* on page 2-85	Master	Timer signals to SOC.
acp_s	PVBus	Slave	AXI ACP slave port.
broadcastatomic	*Signal* on page 2-85	Slave	CHI defined pins.
broadcastcachemaint	*Signal* on page 2-85	Slave	ACE defined pins.
broadcastouter	*Signal* on page 2-85	Slave	ACE defined pins.
broadcastpersist	*Signal* on page 2-85	Slave	CHI defined pins.
cfgend[8]	*Signal* on page 2-85	Slave	This signal if for EE bit initialisation.
cfgsdisable	*Signal* on page 2-85	Slave	This signal disables write access to some secure Interrupt Controller registers.
cfgte[8]	*Signal* on page 2-85	Slave	This signal provides default exception handling state.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-162 Ports (continued)

Name	Protocol	Type	Description
clk_in	ClockSignal	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the cluster level components run e.g cluster level timers, caches and pmu.
clusterid	*Value* on page 2-85	Slave	The port reads the value in CPU ID register field, bits[11:8] of the MPIDR.
clusterpmuirq	*Signal* on page 2-85	Master	DynamIQ pmu irq
cntvalueb	*CounterInterface* on page 2-84	Slave	Interface to SoC level counter module.
commirq[8]	*Signal* on page 2-85	Master	Interrupt signal from debug communications channel.
core_clk_in[8]	ClockSignal	Slave	The clock signal connected to the core_clk_in port is used to determine the rate at which each core executes instructions.
cpuporeset[8]	*Signal* on page 2-85	Slave	CPU power on reset. Initializes all the processor logic, including debug logic.
cryptodisable[8]	*Signal* on page 2-85	Slave	Disable cryptography extensions after reset.
cti[8]	*v8EmbeddedCrossTrigger_controlprotocol* on page 2-84	Master	Cross trigger matrix port.
ctidbgirq[8]	*Signal* on page 2-85	Master	Cross Trigger Interface (CTI) interrupt trigger output.
dbgen	*Signal* on page 2-85	Slave	External debug interface.
dbgnopwrdwn[8]	*Signal* on page 2-85	Master	No power-down request.
dbgpwrupreq[8]	*Signal* on page 2-85	Master	Debug power up request.
dev_debug_s	PVBus	Slave	External debug interface.
event	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE.
fiq[8]	*Signal* on page 2-85	Slave	This signal drives the CPUs fast-interrupt handling.
gicv3_redistributor_s[8]	*GICv3Comms* on page 2-84	Slave	GICv3 AXI-stream port.
irq[8]	*Signal* on page 2-85	Slave	This signal drives the CPUs interrupt handling.
memorymapped_debug_s	PVBus	Slave	External debug interface.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-162 Ports (continued)

Name	Protocol	Type	Description
niden	*Signal* on page 2-85	Slave	External debug interface.
pchannel_cluster	*PChannel* on page 2-80	Slave	PChannel for cluster.
pchannel_core[8]	*PChannel* on page 2-80	Slave	PChannels for cores
pmuirq[8]	*Signal* on page 2-85	Master	Interrupt signal from performance monitoring unit.
presetdbg	*Signal* on page 2-85	Slave	Initialize the shared debug APB, Cross Trigger Interface (CTI), and Cross Trigger Matrix (CTM) logic.
pvbus_m0	PVBus	Master	The core will generate bus requests on this port.
pvbus_periph_m	PVBus	Master	The core can generate peripheral bus request on this port.
rei[8]	*Signal* on page 2-85	Slave	Per core RAM Error Interrupt.
reset[8]	*Signal* on page 2-85	Slave	Raising this signal will put the core into reset mode.
rvbaraddr[8]	*Value_64* on page 2-85	Slave	Reset vector base address.
sci_m	SystemCoherencyInterface	Master	System coherency interface port, which is used to take the whole cluster into/out-of coherency domain
sei[8]	*Signal* on page 2-85	Slave	Per core System Error physical pins.
spiden	*Signal* on page 2-85	Slave	External debug interface.
spniden	*Signal* on page 2-85	Slave	External debug interface.
sporeset	*Signal* on page 2-85	Slave	A single cluster-wide power on reset signal for all resettable registers in DynamIQ.
ticks[8]	*InstructionCount* on page 2-84	Master	This port should be connected to one of the two ticks ports on a 'visualisation' component, in order to display a running instruction count.
vcpumntirq[8]	*Signal* on page 2-85	Master	Interrupt signal for virtual CPU maintenance IRQ.
vfiq[8]	*Signal* on page 2-85	Slave	Virtualised FIQ.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-162 Ports (continued)

Name	Protocol	Type	Description
vinithi[8]	*Signal* on page 2-85	Slave	This signal controls of the location of the exception vectors at reset.
virq[8]	*Signal* on page 2-85	Slave	Virtualised IRQ.
virtio_s	PVBus	Slave	The virtio coherent port, hooks directly into the L2 system and becomes coherent (assuming attributes are set correctly).
vsei[8]	*Signal* on page 2-85	Slave	Per core virtual System Error physical pins.

Table 3-163 Parameters for Cluster_ARM_Cortex-A78C

Name	Type	Default value	Description
BROADCASTATOMIC	bool	0x1	Enable broadcasting of atomic operation. The broadcastatomic signal will override this value if used
BROADCASTCACHEMAINT	bool	0x1	Enable broadcasting of cache maintenance operations to downstream caches. The broadcastcachemaint signal will override this value if used.
BROADCASTOUTER	bool	0x1	Enable broadcasting of Outer Shareable transactions. The broadcastouter signal will override this value if used.
BROADCASTPERSIST	bool	0x1	Enable broadcasting of cache clean to the point of persistence operations. The broadcastpersist signal will override this value if used
CLUSTER_ID	int	0x0	Processor cluster ID value
GICDISABLE	bool	0x1	Disable the new style GICv3 CPU interface in each core model. Should be left enabled unless the platform contains a GICv3 distributor.
cpi_div	int	0x1	Divider for calculating CPI (Cycles Per Instruction)
cpi_mul	int	0x1	Multiplier for calculating CPI (Cycles Per Instruction)
dcache-hit_latency	int	0x0	L1 D-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
dcache-maintenance_latency	int	0x0	L1 D-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-163 Parameters for Cluster_ARM_Cortex-A78C (continued)

Name	Type	Default value	Description
dcache-miss_latency	int	0x0	L1 D-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
dcache-prefetch_enabled	bool	0x0	Enable simulation of data cache prefetching. This is only used when dcache-state_modelled=true
dcache-read_access_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per access (of size dcache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
dcache-read_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per byte accessed.dcache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
dcache-size	int	0x10000	L1 D-Cache size in bytes.
dcache-snoop_data_transfer_latency	int	0x0	L1 D-Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
dcache-state_modelled	bool	0x0	Set whether D-cache has stateful implementation
dcache-write_access_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per access (of size dcache-write_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-write_latency is set. This is only used when dcache-state_modelled=true.
dcache-write_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per byte accessed. dcache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
default_opmode	int	0x4	Operating mode of DynamIQ coming out of reset. 0: SFONLY ON, 1: 1/4 CACHE ON, 2: 1/2 CACHE ON, 3: 3/4 CACHE ON, 4: FULL CACHE ON
diagnostics	bool	0x0	Enable DynamIQ diagnostic messages

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-163 Parameters for Cluster_ARM_Cortex-A78C (continued)

Name	Type	Default value	Description
enable_simulation_performance_optimizations	bool	0x1	With this option enabled, the model will run more quickly, but be less accurate to exact CPU behavior. The model will still be functionally accurate for software, but may increase differences seen between hardware behavior and model behavior for certain workloads (it changes the micro-architectural value of stage12_tlb_size parameter to 1024).
ext_abort_device_read_is_sync	bool	0x0	Synchronous reporting of device-nGnRE read external aborts
ext_abort_device_write_is_sync	bool	0x0	Synchronous reporting of device-nGnRE write external aborts
ext_abort_so_read_is_sync	bool	0x0	Synchronous reporting of device-nGnRnE read external aborts
ext_abort_so_write_is_sync	bool	0x0	Synchronous reporting of device-nGnRnE write external aborts
has_coherent_icache	bool	0x0	Whether icache invalidation to the point of unification is required for instruction to data coherence. true - Invalidate operations not required
has_dot_product	int	0x2	Possible values of this parameter are: - 1, feature not implemented. - 2, feature is implemented.
has_peripheral_port	bool	0x0	If true, additional AXI peripheral port is configured.
icache-hit_latency	int	0x0	L1 I-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when icache-state_modelled=true.
icache-maintenance_latency	int	0x0	L1 I-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when icache-state_modelled=true.
icache-miss_latency	int	0x0	L1 I-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when icache-state_modelled=true.
icache-prefetch_enabled	bool	0x0	Enable simulation of instruction cache prefetching. This is only used when icache-state_modelled=true.
icache-read_access_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per access (of size icache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if icache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when icache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-163 Parameters for Cluster_ARM_Cortex-A78C (continued)

Name	Type	Default value	Description
icache-read_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per byte accessed.icache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when icache-state_modelled=true.
icache-size	int	0x10000	L1 I-Cache size in bytes.
icache-state_modelled	bool	0x0	Set whether I-cache has stateful implementation
l3cache-hit_latency	int	0x0	L3 Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
l3cache-maintenance_latency	int	0x0	L3 Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
l3cache-miss_latency	int	0x0	L3 Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
l3cache-read_access_latency	int	0x0	L3 Cache timing annotation latency for read accesses given in ticks per access (of size l3cache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l3cache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
l3cache-read_latency	int	0x0	L3 Cache timing annotation latency for read accesses given in ticks per byte accessed.l3cache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
l3cache-size	int	0x100000	L3 Cache size in bytes.
l3cache-snoop_data_transfer_latency	int	0x0	L3 Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
l3cache-snoop_issue_latency	int	0x0	L3 Cache timing annotation latency for snoop accesses issued by this cache in total ticks. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-163 Parameters for Cluster_ARM_Cortex-A78C (continued)

Name	Type	Default value	Description
l3cache-write_access_latency	int	0x0	L3 Cache timing annotation latency for write accesses given in ticks per access (of size l3cache-write_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l3cache-write_latency is set. This is only used when dcache-state_modelled=true.
l3cache-write_latency	int	0x0	L3 Cache timing annotation latency for write accesses given in ticks per byte accessed. l3cache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
pchannel_treat_simreset_as_poreset	bool	0x0	Register core as ON state to cluster with simulation reset.
periph_address_end	int	0x0	End address for peripheral port address range exclusive(corresponds to AENDMP input signal).
periph_address_start	int	0x0	Start address for peripheral port address range inclusive(corresponds to ASTARTMP input signal).
ptw_latency	int	0x0	Page table walker latency for TA (Timing Annotation), expressed in simulation ticks
tlb_latency	int	0x0	TLB latency for TA (Timing Annotation), expressed in simulation ticks
treat-dcache-cmos-to-pou-as-nop	bool	0x0	Whether dcache invalidation to the point of unification is required for instruction to data coherence. true - Invalidate operations not required
walk_cache_latency	int	0x0	Walk cache latency for TA (Timing Annotation), expressed in simulation ticks

Table 3-164 Parameters for ARM_Cortex-A78C

Name	Type	Default value	Description
cpu0.CFGEND	bool	0x0	Endianness configuration at reset. 0, little endian. 1, big endian.
cpu0.CFGTE	bool	0x0	Instruction set state when resetting into AArch32. 0, A32. 1, T32.
cpu0.CRYPTODISABLE	bool	0x0	Disable cryptographic features.
cpu0.RVBARADDR	int	0x0	Value of RVBAR_ELx register.
cpu0.VINITHI	bool	0x0	Reset value of SCTLR.V.
cpu0.enable_trace_special_hlt_imm16	bool	0x0	Enable usage of parameter trace_special_hlt_imm16
cpu0.l2cache-hit_latency	int	0x0	L2 Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-164 Parameters for ARM_Cortex-A78C (continued)

Name	Type	Default value	Description
cpu0.l2cache-maintenance_latency	int	0x0	L2 Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
cpu0.l2cache-miss_latency	int	0x0	L2 Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
cpu0.l2cache-read_access_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
cpu0.l2cache-read_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per byte accessed.l2cache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
cpu0.l2cache-size	int	0x80000	L2 Cache size in bytes.
cpu0.l2cache-snoop_data_transfer_latency	int	0x0	L2 Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
cpu0.l2cache-snoop_issue_latency	int	0x0	L2 Cache timing annotation latency for snoop accesses issued by this cache in total ticks. This is only used when dcache-state_modelled=true.
cpu0.l2cache-write_access_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-write_latency is set. This is only used when dcache-state_modelled=true.
cpu0.l2cache-write_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per byte accessed. l2cache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
cpu0.max_code_cache_mb	int	0x100	Maximum size of the simulation code cache (MiB). For platforms with more than 2 cores this limit will be scaled down. (e.g 1/8 for 16 or more cores)
cpu0.min_sync_level	int	0x0	Force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-164 Parameters for ARM_Cortex-A78C (continued)

Name	Type	Default value	Description
cpu0.semihosting-A32_HLT	int	0xf000	A32 HLT number for semihosting calls.
cpu0.semihosting-A64_HLT	int	0xf000	A64 HLT number for semihosting calls.
cpu0.semihosting-ARM_SVC	int	0x123456	A32 SVC number for semihosting calls.
cpu0.semihosting-T32_HLT	int	0x3c	T32 HLT number for semihosting calls.
cpu0.semihosting-Thumb_SVC	int	0xab	T32 SVC number for semihosting calls.
cpu0.semihosting-cmd_line	string	""	Command line available to semihosting calls.
cpu0.semihosting-cwd	string	""	Base directory for semihosting file access.
cpu0.semihosting-enable	bool	0x1	Enable semihosting SVC/HLT traps.
cpu0.semihosting-heap_base	int	0x0	Virtual address of heap base.
cpu0.semihosting-heap_limit	int	0xf000000	Virtual address of top of heap.
cpu0.semihosting-stack_base	int	0x10000000	Virtual address of base of descending stack.
cpu0.semihosting-stack_limit	int	0xf000000	Virtual address of stack limit.
cpu0.trace_special_hlt_imm16	int	0xf000	For this HLT number, IF enable_trace_special_hlt_imm16=true, skip performing usual HLT execution but call MTI trace if registered
cpu0.vfp-enable_at_reset	bool	0x0	Enable VFP in CPACR, CPPWR, NSACR at reset. Warning: Arm recommends going through the implementation's suggested VFP power-up sequence!

3.4.30 ARMCortexA78CT

ARMCortexA78CT CPU component. This model is written in C++ and models version r0p0 of the RTL.

ARMCortexA78CT contains the following CADI targets:

- ARM_Cortex-A78
- Cluster_ARM_Cortex-A78
- PVCache
- TlbCadi

ARMCortexA78CT contains the following MTI components:

- *4.24 ARM_Cortex-A78 - trace* on page 4-1634
- *4.48 ARMv8Cluster - trace* on page 4-2401
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402
- *4.68 DSU - trace* on page 4-2463
- *4.73 GICv3CPUInterfaceDecoder - trace* on page 4-2489
- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.108 PVCache - trace* on page 4-2761
- *4.114 TLB - trace* on page 4-2834

About ARMCortexA78CT

The model supports the following features:

- DynamIQ (FCM) system registers.
- Per-core L2 cache.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

- A P-Channel for the cluster and for each core.
- Optional peripheral port.
- L3 cache partition.
- Per-core clock.
- BROADCASTPERSIST pin.

The model does not support the following features:

- BROADCASTCACHEMAINTPOU pin.
- COREINSTRRET, COREINSTRRUN, or nPMBIRQ signals.
- DynamIQ features that are irrelevant to the programmers' view simulation, for example:
  — Automatic CPU retention mode.
  — Level-3 Cache RAM retention.
- 256-bit wide output transactions.
- ECC and parity schemes are hardware-specific so are not supported.
- Self-test features (MBIST).
- Latency configuration.
- Snoop filtering.
- Debug power management signals DBGPWRDUP and DBGRSTREQ are not implemented, but DBGPWRUPREQ and DBGNOPWRDWN are implemented.
- Cache stashing capability.
- Each thread has its own L1Cache and L2Cache instead of a common cache that is shared by all threads of the core.
- Per-thread parameters are not yet supported but signals are implemented.

This component has a variable number of cores per cluster, specified using the NUM_CORES parameter. The following components also exist, with a fixed number of cores per cluster:

- ARMCortexA78x1CT.
- ARMCortexA78x2CT.
- ARMCortexA78x3CT.
- ARMCortexA78x4CT.

The per-core parameters are preceded by cpu*n*., where *n* identifies the core (0-3).

Table 3-165 Ports

Name	Protocol	Type	Description
AENDMP	*Value_64* on page 2-85	Slave	Port to obtain end address of valid peripheral address range (exclusive).
ASTARTMP	*Value_64* on page 2-85	Slave	Port to obtain start address of valid peripheral address range (inclusive).
CNTHPIRQ[4]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTHVIRQ[4]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTPNSIRQ[4]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTPSIRQ[4]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTVIRQ[4]	*Signal* on page 2-85	Master	Timer signals to SOC.
acp_s	PVBus	Slave	AXI ACP slave port.
broadcastatomic	*Signal* on page 2-85	Slave	CHI defined pins.
broadcastcachemaint	*Signal* on page 2-85	Slave	ACE defined pins.
broadcastouter	*Signal* on page 2-85	Slave	ACE defined pins.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-165 Ports (continued)

Name	Protocol	Type	Description
broadcastpersist	*Signal* on page 2-85	Slave	CHI defined pins.
cfgend[4]	*Signal* on page 2-85	Slave	This signal if for EE bit initialisation.
cfgsdisable	*Signal* on page 2-85	Slave	This signal disables write access to some secure Interrupt Controller registers.
cfgte[4]	*Signal* on page 2-85	Slave	This signal provides default exception handling state.
clk_in	ClockSignal	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the cluster level components run e.g cluster level timers, caches and pmu.
clusterid	*Value* on page 2-85	Slave	The port reads the value in CPU ID register field, bits[11:8] of the MPIDR.
clusterpmuirq	*Signal* on page 2-85	Master	DynamIQ pmu irq
cntvalueb	*CounterInterface* on page 2-84	Slave	Interface to SoC level counter module.
commirq[4]	*Signal* on page 2-85	Master	Interrupt signal from debug communications channel.
core_clk_in[4]	ClockSignal	Slave	The clock signal connected to the core_clk_in port is used to determine the rate at which each core executes instructions.
cpuporeset[4]	*Signal* on page 2-85	Slave	CPU power on reset. Initializes all the processor logic, including debug logic.
cryptodisable[4]	*Signal* on page 2-85	Slave	Disable cryptography extensions after reset.
cti[4]	*v8EmbeddedCrossTrigger_controlprotocol* on page 2-84	Master	Cross trigger matrix port.
ctidbgirq[4]	*Signal* on page 2-85	Master	Cross Trigger Interface (CTI) interrupt trigger output.
dbgen[4]	*Signal* on page 2-85	Slave	External debug interface.
dbgnopwrdwn[4]	*Signal* on page 2-85	Master	No power-down request.
dbgpwrupreq[4]	*Signal* on page 2-85	Master	Debug power up request.
dev_debug_s	PVBus	Slave	External debug interface.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-165 Ports (continued)

Name	Protocol	Type	Description
event	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE.
fiq[4]	*Signal* on page 2-85	Slave	This signal drives the CPUs fast-interrupt handling.
gicv3_redistributor_s[4]	*GICv3Comms* on page 2-84	Slave	GICv3 AXI-stream port.
irq[4]	*Signal* on page 2-85	Slave	This signal drives the CPUs interrupt handling.
memorymapped_debug_s	PVBus	Slave	External debug interface.
niden[4]	*Signal* on page 2-85	Slave	External debug interface.
pchannel_cluster	*PChannel* on page 2-80	Slave	PChannel for cluster.
pchannel_core[4]	*PChannel* on page 2-80	Slave	PChannels for cores
pmuirq[4]	*Signal* on page 2-85	Master	Interrupt signal from performance monitoring unit.
presetdbg	*Signal* on page 2-85	Slave	Initialize the shared debug APB, Cross Trigger Interface (CTI), and Cross Trigger Matrix (CTM) logic.
pvbus_m0	PVBus	Master	The core will generate bus requests on this port.
pvbus_periph_m	PVBus	Master	The core can generate peripheral bus request on this port.
rei[4]	*Signal* on page 2-85	Slave	Per core RAM Error Interrupt.
reset[4]	*Signal* on page 2-85	Slave	Raising this signal will put the core into reset mode.
rvbaraddr[4]	*Value_64* on page 2-85	Slave	Reset vector base address.
sci_m	SystemCoherencyInterface	Master	System coherency interface port, which is used to take the whole cluster into/out-of coherency domain
sei[4]	*Signal* on page 2-85	Slave	Per core System Error physical pins.
spiden[4]	*Signal* on page 2-85	Slave	External debug interface.
spniden[4]	*Signal* on page 2-85	Slave	External debug interface.
sporeset	*Signal* on page 2-85	Slave	A single cluster-wide power on reset signal for all resettable registers in DynamIQ.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-165 Ports (continued)

Name	Protocol	Type	Description
ticks[4]	*InstructionCount* on page 2-84	Master	This port should be connected to one of the two ticks ports on a 'visualisation' component, in order to display a running instruction count.
vcpumntirq[4]	*Signal* on page 2-85	Master	Interrupt signal for virtual CPU maintenance IRQ.
vfiq[4]	*Signal* on page 2-85	Slave	Virtualised FIQ.
vinithi[4]	*Signal* on page 2-85	Slave	This signal controls of the location of the exception vectors at reset.
virq[4]	*Signal* on page 2-85	Slave	Virtualised IRQ.
virtio_s	PVBus	Slave	The virtio coherent port, hooks directly into the L2 system and becomes coherent (assuming attributes are set correctly).
vsei[4]	*Signal* on page 2-85	Slave	Per core virtual System Error physical pins.

Table 3-166 Parameters for Cluster_ARM_Cortex-A78

Name	Type	Default value	Description
BROADCASTATOMIC	bool	0x1	Enable broadcasting of atomic operation. The broadcastatomic signal will override this value if used
BROADCASTCACHEMAINT	bool	0x1	Enable broadcasting of cache maintenance operations to downstream caches. The broadcastcachemaint signal will override this value if used.
BROADCASTOUTER	bool	0x1	Enable broadcasting of Outer Shareable transactions. The broadcastouter signal will override this value if used.
BROADCASTPERSIST	bool	0x1	Enable broadcasting of cache clean to the point of persistence operations. The broadcastpersist signal will override this value if used
CLUSTER_ID	int	0x0	Processor cluster ID value
GICDISABLE	bool	0x1	Disable the new style GICv3 CPU interface in each core model. Should be left enabled unless the platform contains a GICv3 distributor.
cpi_div	int	0x1	Divider for calculating CPI (Cycles Per Instruction)
cpi_mul	int	0x1	Multiplier for calculating CPI (Cycles Per Instruction)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-166 Parameters for Cluster_ARM_Cortex-A78 (continued)

Name	Type	Default value	Description
dcache-hit_latency	int	0x0	L1 D-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
dcache-maintenance_latency	int	0x0	L1 D-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
dcache-miss_latency	int	0x0	L1 D-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
dcache-prefetch_enabled	bool	0x0	Enable simulation of data cache prefetching. This is only used when dcache-state_modelled=true
dcache-read_access_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per access (of size dcache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
dcache-read_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per byte accessed.dcache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
dcache-size	int	0x10000	L1 D-Cache size in bytes.
dcache-snoop_data_transfer_latency	int	0x0	L1 D-Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
dcache-state_modelled	bool	0x0	Set whether D-cache has stateful implementation
dcache-write_access_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per access (of size dcache-write_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-write_latency is set. This is only used when dcache-state_modelled=true.
dcache-write_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per byte accessed. dcache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-166 Parameters for Cluster_ARM_Cortex-A78 (continued)

Name	Type	Default value	Description
default_opmode	int	0x4	Operating mode of DynamIQ coming out of reset. 0: SFONLY ON, 1: 1/4 CACHE ON, 2: 1/2 CACHE ON, 3: 3/4 CACHE ON, 4: FULL CACHE ON
diagnostics	bool	0x0	Enable DynamIQ diagnostic messages
enable_simulation_performance_optimizations	bool	0x1	With this option enabled, the model will run more quickly, but be less accurate to exact CPU behavior. The model will still be functionally accurate for software, but may increase differences seen between hardware behavior and model behavior for certain workloads (it changes the micro-architectural value of stage12_tlb_size parameter to 1024).
ext_abort_device_read_is_sync	bool	0x0	Synchronous reporting of device-nGnRE read external aborts
ext_abort_device_write_is_sync	bool	0x0	Synchronous reporting of device-nGnRE write external aborts
ext_abort_so_read_is_sync	bool	0x0	Synchronous reporting of device-nGnRnE read external aborts
ext_abort_so_write_is_sync	bool	0x0	Synchronous reporting of device-nGnRnE write external aborts
has_coherent_icache	bool	0x0	Whether icache invalidation to the point of unification is required for instruction to data coherence. true - Invalidate operations not required
has_dot_product	int	0x2	Possible values of this parameter are: - 1, feature not implemented. - 2, feature is implemented.
has_peripheral_port	bool	0x0	If true, additional AXI peripheral port is configured.
icache-hit_latency	int	0x0	L1 I-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when icache-state_modelled=true.
icache-maintenance_latency	int	0x0	L1 I-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when icache-state_modelled=true.
icache-miss_latency	int	0x0	L1 I-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when icache-state_modelled=true.
icache-prefetch_enabled	bool	0x0	Enable simulation of instruction cache prefetching. This is only used when icache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-166 Parameters for Cluster_ARM_Cortex-A78 (continued)

Name	Type	Default value	Description
icache-read_access_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per access (of size icache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if icache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when icache-state_modelled=true.
icache-read_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per byte accessed.icache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when icache-state_modelled=true.
icache-size	int	0x10000	L1 I-Cache size in bytes.
icache-state_modelled	bool	0x0	Set whether I-cache has stateful implementation
l3cache-hit_latency	int	0x0	L3 Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
l3cache-maintenance_latency	int	0x0	L3 Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
l3cache-miss_latency	int	0x0	L3 Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
l3cache-read_access_latency	int	0x0	L3 Cache timing annotation latency for read accesses given in ticks per access (of size l3cache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l3cache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
l3cache-read_latency	int	0x0	L3 Cache timing annotation latency for read accesses given in ticks per byte accessed.l3cache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-166 Parameters for Cluster_ARM_Cortex-A78 (continued)

Name	Type	Default value	Description
l3cache-size	int	0x100000	L3 Cache size in bytes.
l3cache-snoop_data_transfer_latency	int	0x0	L3 Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
l3cache-snoop_issue_latency	int	0x0	L3 Cache timing annotation latency for snoop accesses issued by this cache in total ticks. This is only used when dcache-state_modelled=true.
l3cache-write_access_latency	int	0x0	L3 Cache timing annotation latency for write accesses given in ticks per access (of size l3cache-write_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l3cache-write_latency is set. This is only used when dcache-state_modelled=true.
l3cache-write_latency	int	0x0	L3 Cache timing annotation latency for write accesses given in ticks per byte accessed. l3cache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
pchannel_treat_simreset_as_poreset	bool	0x0	Register core as ON state to cluster with simulation reset.
periph_address_end	int	0x0	End address for peripheral port address range exclusive(corresponds to AENDMP input signal).
periph_address_start	int	0x0	Start address for peripheral port address range inclusive(corresponds to ASTARTMP input signal).
ptw_latency	int	0x0	Page table walker latency for TA (Timing Annotation), expressed in simulation ticks
tlb_latency	int	0x0	TLB latency for TA (Timing Annotation), expressed in simulation ticks
treat-dcache-cmos-to-pou-as-nop	bool	0x0	Whether dcache invalidation to the point of unification is required for instruction to data coherence. true - Invalidate operations not required
walk_cache_latency	int	0x0	Walk cache latency for TA (Timing Annotation), expressed in simulation ticks

Table 3-167 Parameters for ARM_Cortex-A78

Name	Type	Default value	Description
cpu0.CFGEND	bool	0x0	Endianness configuration at reset. 0, little endian. 1, big endian.
cpu0.CFGTE	bool	0x0	Instruction set state when resetting into AArch32. 0, A32. 1, T32.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-167 Parameters for ARM_Cortex-A78 (continued)

Name	Type	Default value	Description
cpu0.CRYPTODISABLE	bool	0x0	Disable cryptographic features.
cpu0.RVBARADDR	int	0x0	Value of RVBAR_ELx register.
cpu0.VINITHI	bool	0x0	Reset value of SCTLR.V.
cpu0.enable_trace_special_hlt_imm16	bool	0x0	Enable usage of parameter trace_special_hlt_imm16
cpu0.l2cache-hit_latency	int	0x0	L2 Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
cpu0.l2cache-maintenance_latency	int	0x0	L2 Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
cpu0.l2cache-miss_latency	int	0x0	L2 Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
cpu0.l2cache-read_access_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
cpu0.l2cache-read_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per byte accessed.l2cache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
cpu0.l2cache-size	int	0x80000	L2 Cache size in bytes.
cpu0.l2cache-snoop_data_transfer_latency	int	0x0	L2 Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
cpu0.l2cache-snoop_issue_latency	int	0x0	L2 Cache timing annotation latency for snoop accesses issued by this cache in total ticks. This is only used when dcache-state_modelled=true.
cpu0.l2cache-write_access_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-write_latency is set. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-167 Parameters for ARM_Cortex-A78 (continued)

Name	Type	Default value	Description
cpu0.l2cache-write_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per byte accessed. l2cache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
cpu0.max_code_cache_mb	int	0x100	Maximum size of the simulation code cache (MiB). For platforms with more than 2 cores this limit will be scaled down. (e.g 1/8 for 16 or more cores)
cpu0.min_sync_level	int	0x0	Force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)
cpu0.semihosting-A32_HLT	int	0xf000	A32 HLT number for semihosting calls.
cpu0.semihosting-A64_HLT	int	0xf000	A64 HLT number for semihosting calls.
cpu0.semihosting-ARM_SVC	int	0x123456	A32 SVC number for semihosting calls.
cpu0.semihosting-T32_HLT	int	0x3c	T32 HLT number for semihosting calls.
cpu0.semihosting-Thumb_SVC	int	0xab	T32 SVC number for semihosting calls.
cpu0.semihosting-cmd_line	string	""	Command line available to semihosting calls.
cpu0.semihosting-cwd	string	""	Base directory for semihosting file access.
cpu0.semihosting-enable	bool	0x1	Enable semihosting SVC/HLT traps.
cpu0.semihosting-heap_base	int	0x0	Virtual address of heap base.
cpu0.semihosting-heap_limit	int	0xf000000	Virtual address of top of heap.
cpu0.semihosting-stack_base	int	0x10000000	Virtual address of base of descending stack.
cpu0.semihosting-stack_limit	int	0xf000000	Virtual address of stack limit.
cpu0.trace_special_hlt_imm16	int	0xf000	For this HLT number, IF enable_trace_special_hlt_imm16=true, skip performing usual HLT execution but call MTI trace if registered
cpu0.vfp-enable_at_reset	bool	0x0	Enable VFP in CPACR, CPPWR, NSACR at reset. Warning: Arm recommends going through the implementation's suggested VFP power-up sequence!

3.4.31 ARMCortexM0CT

ARMCortexM0CT CPU component. This model is written in C++ and models version r0p0 of the RTL.

ARMCortexM0CT contains the following CADI targets:

- ARM_Cortex-M0

ARMCortexM0CT contains the following MTI components:

- *4.29 ARM_Cortex-M0 - trace* on page 4-1808
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402
- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Differences between the model and the RTL

- This model does not have a parameter that is equivalent to the RAR integration option. The architecturally required register state is reset.
- This model exposes, through CADI, a VTOR register, but this register in not present in hardware.
- Arm does not guarantee that all Armv7-M behavior is absent from models of Armv6-M cores. As a consequence, Arm does not guarantee that code that runs on Armv7-M cores but fails on Armv6-M cores also fails on Armv6-M Fast Models cores.

Table 3-168 Ports

Name	Protocol	Type	Description
ahbd	PVBus	Slave	Debug AHB - core bus slave driven by the DAP.
bigend	*Signal* on page 2-85	Slave	Configure big endian data format.
clk_in	ClockSignal	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the core executes instructions.
currpri	*Value* on page 2-85	Master	Current execution priority.
edbgrq	*Signal* on page 2-85	Slave	External debug request.
event	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE and corresponds to the RTL TXEV and RXEV signals.
intisr[32]	*Signal* on page 2-85	Slave	This signal array delivers signals to the NVIC.
intnmi	*Signal* on page 2-85	Slave	Configure non maskable interrupt.
lockup	*Signal* on page 2-85	Master	Asserted when the processor is in lockup state.
poreset	*Signal* on page 2-85	Slave	Raising this signal will do a power-on reset of the core.
pv_ppbus_m	PVBus	Master	The core will generate External Private Peripheral Bus requests on this port.
pvbus_m	PVBus	Master	The core will generate bus requests on this port.
sleepdeep	*Signal* on page 2-85	Master	Asserted when the processor is in deep sleep.
sleeping	*Signal* on page 2-85	Master	Asserted when the processor is in sleep.
stcalib	*Value* on page 2-85	Slave	This is the calibration value for the SysTick timer.
stclk	ClockSignal	Slave	This is the reference clock for the SysTick timer.
sysreset	*Signal* on page 2-85	Slave	Raising this signal will put the core into reset mode (but does not reset the debug logic).
sysresetreq	*Signal* on page 2-85	Master	Asserted to indicate that a reset is required.
ticks	*InstructionCount* on page 2-84	Master	Port allowing the number of instructions since startup to be read from the CPU.

Table 3-169 Parameters for ARM_Cortex-M0

Name	Type	Default value	Description
BIGENDINIT	bool	0x0	Initialize processor to big endian mode
BKPT	int	0x4	Number of breakpoint unit comparators implemented
DBG	bool	0x1	Set whether debug extensions are implemented
NUM_IRQ	int	0x20	Number of user interrupts

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-169 Parameters for ARM_Cortex-M0 (continued)

Name	Type	Default value	Description
SYST	bool	0x1	Enable support for SysTick timer functionality
WIC	bool	0x1	Include support for WIC-mode deep sleep
WPT	int	0x2	Number of watchpoint unit comparators implemented
cpi_div	int	0x1	divider for calculating CPI (Cycles Per Instruction)
cpi_mul	int	0x1	multiplier for calculating CPI (Cycles Per Instruction)
master_id	int	0x0	Master ID presented in bus transactions
min_sync_level	int	0x0	force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)
semihosting-Thumb_SVC	int	0xab	T32 SVC number for semihosting
semihosting-cmd_line	string	""	Command line available to semihosting SVC calls
semihosting-cwd	string	""	Base directory for semihosting file access.
semihosting-enable	bool	0x1	Enable semihosting SVC traps. Applications that do not use semihosting must set this parameter to false.
semihosting-heap_base	int	0x0	Virtual address of heap base
semihosting-heap_limit	int	0x10700000	Virtual address of top of heap
semihosting-prefix	bool	0x0	Prefix semihosting output with target instance name
semihosting-stack_base	int	0x10700000	Virtual address of base of descending stack
semihosting-stack_limit	int	0x10800000	Virtual address of stack limit

3.4.32 ARMCortexM0PlusCT

ARMCortexM0PlusCT CPU component. This model is written in C++ and models version r0p1 of the RTL.

ARMCortexM0PlusCT contains the following CADI targets:

- ARM_Cortex-M0+

ARMCortexM0PlusCT contains the following MTI components:

- *4.30 ARM_Cortex-M0+ - trace* on page 4-1839
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402
- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758

Differences between the model and the RTL

- This model does not have a parameter that is equivalent to the RAR integration option. The architecturally required register state is reset.
- Arm does not guarantee that all Armv7-M behavior is absent from models of Armv6-M cores. As a consequence, Arm does not guarantee that code that runs on Armv7-M cores but fails on Armv6-M cores also fails on Armv6-M Fast Models cores.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-170 Ports

Name	Protocol	Type	Description
ahbd	PVBus	Slave	Debug AHB - core bus slave driven by the DAP.
bigend	*Signal* on page 2-85	Slave	Configure big endian data format.
clk_in	ClockSignal	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the core executes instructions.
currpri	*Value* on page 2-85	Master	Current execution priority.
edbgrq	*Signal* on page 2-85	Slave	External debug request.
event	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE and corresponds to the RTL TXEV and RXEV signals.
intisr[32]	*Signal* on page 2-85	Slave	This signal array delivers signals to the NVIC.
intnmi	*Signal* on page 2-85	Slave	Configure non maskable interrupt.
io_port_in	PVBus	Slave	I/O port pair. See the documentation for the io_port_out port.
io_port_out	PVBus	Master	I/O port pair. Used if IOP is true. Transactions from io_port_out which do not "match" should be returned via io_port_in. For performance reasons, the I/O port interface is not modelled directly. Instead, a simple PVBus gasket is inserted at the point in the memory system where the I/O port would be. In hardware, a device would be attached to the port which would tell the CPU whether it would like to intercept each transaction, given its address. This can be modelled in a performant manner by connecting a PVBusMapper-based device to io_port_out which intercepts transactions of interest and passes other transactions back to the CPU via io_port_in. Your I/O port device model is also responsible for aborting transactions which would be aborted on hardware (e.g. exclusives) if necessary.
lockup	*Signal* on page 2-85	Master	Asserted when the processor is in lockup state.
poreset	*Signal* on page 2-85	Slave	Raising this signal will do a power-on reset of the core.
pv_ppbus_m	PVBus	Master	The core will generate External Private Peripheral Bus requests on this port.
pvbus_m	PVBus	Master	The core will generate bus requests on this port.
sleepdeep	*Signal* on page 2-85	Master	Asserted when the processor is in deep sleep.
sleeping	*Signal* on page 2-85	Master	Asserted when the processor is in sleep.
stcalib	*Value* on page 2-85	Slave	This is the calibration value for the SysTick timer.
stclk	ClockSignal	Slave	This is the reference clock for the SysTick timer.
sysreset	*Signal* on page 2-85	Slave	Raising this signal will put the core into reset mode (but does not reset the debug logic).
sysresetreq	*Signal* on page 2-85	Master	Asserted to indicate that a reset is required.
ticks	*InstructionCount* on page 2-84	Master	Port allowing the number of instructions since startup to be read from the CPU.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-171 Parameters for ARM_Cortex-M0plus

Name	Type	Default value	Description
BIGENDINIT	bool	0x0	Initialize processor to big endian mode
BKPT	int	0x4	Number of breakpoint unit comparators implemented
DBG	bool	0x1	Set whether debug extensions are implemented
IOP	bool	0x0	Send all d-side transactions to the port, io_port_out. Transactions which do not match should be returned to the port, io_port_in
IRQDIS	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n]
NUM_IRQ	int	0x20	Number of user interrupts
NUM_MPU_REGION	int	0x0	Number of MPU regions
SYST	bool	0x1	Enable support for SysTick timer functionality
USER	bool	0x1	Enable support for Unprivileged/Privileged Extension
VTOR	bool	0x1	Include Vector Table Offset Register
WIC	bool	0x1	Include support for WIC-mode deep sleep
WPT	int	0x2	Number of watchpoint unit comparators implemented
cpi_div	int	0x1	divider for calculating CPI (Cycles Per Instruction)
cpi_mul	int	0x1	multiplier for calculating CPI (Cycles Per Instruction)
master_id	int	0x0	Master ID presented in bus transactions
min_sync_level	int	0x0	force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)
scheduler_mode	int	0x0	Control the interleaving of instructions in this processor (0=default long quantum, 1=low latency mode, short quantum and signal checking, 2=lock-breaking mode, long quantum with additional context switches near load-exclusive instructions, 3=ISSCompare)
semihosting-Thumb_SVC	int	0xab	T32 SVC number for semihosting
semihosting-cmd_line	string	""	Command line available to semihosting SVC calls
semihosting-cwd	string	""	Base directory for semihosting file access.
semihosting-enable	bool	0x1	Enable semihosting SVC traps. Applications that do not use semihosting must set this parameter to false.
semihosting-heap_base	int	0x0	Virtual address of heap base
semihosting-heap_limit	int	0x10700000	Virtual address of top of heap
semihosting-prefix	bool	0x0	Prefix semihosting output with target instance name
semihosting-stack_base	int	0x10700000	Virtual address of base of descending stack
semihosting-stack_limit	int	0x10800000	Virtual address of stack limit

3.4.33 ARMCortexM3CT

ARMCortexM3CT CPU component. This model is written in C++ and models version r2p1 of the RTL.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ARMCortexM3CT contains the following CADI targets:

- ARM_Cortex-M3

ARMCortexM3CT contains the following MTI components:

- *4.32 ARM_Cortex-M3 - trace* on page 4-1901
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402
- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758

Differences between the model and the RTL

This component has the following differences from the corresponding revision of the RTL implementation:

- The WIC is not currently implemented.
- Power control is not implemented, so the processor does not set the SLEEPING or SLEEPDEEP signals. It does not support powering down of the processor.
- Only the minimal level of debug support is provided (no DAP, FPB, DWT, or halting debug capability).
- Debug-related components are not implemented.
- The unimplemented registers are the processor debug registers, system debug registers, debug interface port registers, TPIU registers, and ETM registers.
- The processor must still be clocked even if it has asserted the sleeping or sleepdeep signals.
- Disabling processor features using the Auxiliary Control Register is not supported.
- Only a single `pvbus_m` master port is provided. This combines the ICode, DCode, and System bus interfaces of the RTL. The external PPB bus is provided by the `pv_ppbus_m` master port.
- In privileged mode, STRT and LDRT to the PPB region are not forbidden access.
- No support for ETM, TPIU, or HTM.
- There is no supported equivalent of the RESET_ALL_REGS configuration setting in RTL (that forces all registers to have a well-defined value on reset).
- The RTL implements the ROM table as an external component on the External Private Peripheral Bus. In the CT model, the ROM table is implemented internally as a fallback if an external PPB access in the ROM table address region aborts. This permits the default ROM table to be overridden (by implementing an external component connected to the external PPB to handle accesses to these addresses) without requiring every user of the processor to implement and connect a ROM table component.

Implementation of ITM in M-class models

This model has a parameter that enables partial support for Instrumentation Trace Macrocell (ITM). In hardware or RTL, trace data from ITM is sent in packets to the trace block serially using a single pin or wire. In the model, if it is enabled, the ITM trace data is output using an MTI trace source called `ITM`. The `ITM` trace source has an `ITM_PACKET_TYPE` field. The following table shows which packet types the model supports:

Table 3-172 ITM_PACKET_TYPE field values that the model supports

Field value	Description	Supported by model
`ITM_SYNC`	Synchronization packet	Not supported.
`ITM_P_OVERFLOW`	Protocol: Overflow packet	Not supported.
`ITM_P_LOCAL_TIMESTAMP`	Protocol: Local timestamp packets	Not supported.
`ITM_P_GLOBAL_TIMESTAMP`	Protocol: Global timestamp packets	Not supported.
`ITM_P_EXTEN`	Protocol: Extension packet	Not supported.
`ITM_S_INSTRUMENTATION`	Source: Instrumentation packet	Supported.
`ITM_S_DWT_EVENT_COUNTER`	Hardware source: Event counter wrapping	Not supported.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-172 ITM_PACKET_TYPE field values that the model supports (continued)

Field value	Description	Supported by model
ITM_S_DWT_EXCEPTION	Hardware source: Exception tracing	Supported.
ITM_S_DWT_PC_SAMPLING	Hardware source: PC sampling	Not supported.
ITM_S_DWT_DATA_PC_TRACE	Hardware source: DWT Data trace PC value	Supported.
ITM_S_DWT_DATA_ADDRESS_TRACE	Hardware source: DWT Data trace address value	Supported.
ITM_S_DWT_DATA_DATA_TRACE	Hardware source: DWT Data trace DATA value	Supported.

Table 3-173 Ports

Name	Protocol	Type	Description
ahb_ap	PVBus	Slave	Debug AHB - core bus slave driven by the DAP.
auxfault	*Value* on page 2-85	Slave	This is wired to the Auxiliary Fault Status Register.
bigend	*Signal* on page 2-85	Slave	Configure big endian data format.
clk_in	ClockSignal	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the core executes instructions.
currpri	*Value* on page 2-85	Master	Current execution priority.
edbgrq	*Signal* on page 2-85	Slave	External debug request.
event	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE and corresponds to the RTL TXEV and RXEV signals.
intisr[240]	*Signal* on page 2-85	Slave	This signal array delivers signals to the NVIC.
intnmi	*Signal* on page 2-85	Slave	Configure non maskable interrupt.
lockup	*Signal* on page 2-85	Master	Asserted when the processor is in lockup state.
poreset	*Signal* on page 2-85	Slave	Raising this signal will do a power-on reset of the core.
pv_ppbus_m	PVBus	Master	The core will generate External Private Peripheral Bus requests on this port.
pvbus_m	PVBus	Master	The core will generate bus requests on this port.
sleepdeep	*Signal* on page 2-85	Master	Asserted when the processor is in deep sleep.
sleeping	*Signal* on page 2-85	Master	Asserted when the processor is in sleep.
stcalib	*Value* on page 2-85	Slave	This is the calibration value for the SysTick timer.
stclk	ClockSignal	Slave	This is the reference clock for the SysTick timer.
sysreset	*Signal* on page 2-85	Slave	Raising this signal will put the core into reset mode (but does not reset the debug logic).
sysresetreq	*Signal* on page 2-85	Master	Asserted to indicate that a reset is required.
ticks	*InstructionCount* on page 2-84	Master	Port allowing the number of instructions since startup to be read from the CPU.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-174 Parameters for ARM_Cortex-M3

Name	Type	Default value	Description
BB_PRESENT	bool	0x1	Enable bitbanding
BIGENDINIT	bool	0x0	Initialize processor to big endian mode
DBGLVL	int	0x3	0: No debug; 1: Minimal debug; 2: Full debug without DWT address matching; 3: Full debug support with DWT data-comparators
LVL_WIDTH	int	0x3	Number of bits of interrupt priority
NUM_IRQ	int	0x10	Number of user interrupts
NUM_MPU_REGION	int	0x8	Number of MPU regions
TRACE_LVL	bool	0x1	Level of instrumentation trace supported. false : No ITM trace included, true: ITM trace included
WIC	bool	0x1	Include support for WIC-mode deep sleep
cpi_div	int	0x1	divider for calculating CPI (Cycles Per Instruction)
cpi_mul	int	0x1	multiplier for calculating CPI (Cycles Per Instruction)
master_id	int	0x0	Master ID presented in bus transactions
min_sync_level	int	0x0	force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)
scheduler_mode	int	0x0	Control the interleaving of instructions in this processor (0=default long quantum, 1=low latency mode, short quantum and signal checking, 2=lock-breaking mode, long quantum with additional context switches near load-exclusive instructions, 3=ISSCompare)
semihosting-Thumb_SVC	int	0xab	T32 SVC number for semihosting
semihosting-cmd_line	string	""	Command line available to semihosting SVC calls
semihosting-cwd	string	""	Base directory for semihosting file access.
semihosting-enable	bool	0x1	Enable semihosting SVC traps. Applications that do not use semihosting must set this parameter to false.
semihosting-heap_base	int	0x0	Virtual address of heap base
semihosting-heap_limit	int	0x10700000	Virtual address of top of heap
semihosting-prefix	bool	0x0	Prefix semihosting output with target instance name
semihosting-stack_base	int	0x10700000	Virtual address of base of descending stack
semihosting-stack_limit	int	0x10800000	Virtual address of stack limit

3.4.34 ARMCortexM4CT

ARMCortexM4CT CPU component. This model is written in C++ and models version r0p0 of the RTL.

ARMCortexM4CT contains the following CADI targets:

- ARM_Cortex-M4

ARMCortexM4CT contains the following MTI components:

- *4.35 ARM_Cortex-M4 - trace* on page 4-1987
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402
- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Differences between the model and the RTL

This component has the following differences from the corresponding revision of the RTL implementation:

- The *Wakeup Interrupt Controller* (WIC) is not implemented.
- Power control is not implemented. Powering down of the processor is not supported. The processor must still be clocked even if it has asserted the sleeping or sleepdeep signals.
- Only the minimal level of debug support is provided (no DAP, FPB, DWT, or halting debug capability).
- No debug-related components are implemented.
- The unimplemented registers are the processor debug registers, system debug registers, debug interface port registers, TPIU registers, and ETM registers.
- No support for ETM, TPIU, or HTM.
- There is no supported equivalent of the RESET_ALL_REGS configuration setting in RTL (that forces all registers to have a well-defined value on reset).
- Disabling processor features using the Auxiliary Control Register is not supported.
- Only a single `pvbus_m` master port is provided. This combines the ICode, DCode, and System bus interfaces of the RTL. The external PPB bus is provided by the `pv_ppbus_m` master port.
- In privileged mode, STRT and LDRT to the PPB region are not forbidden access.
- The RTL implements the ROM table as an external component on the External Private Peripheral Bus. In the CT model, the ROM table is implemented internally as a fallback if an external PPB access in the ROM table address region aborts. This permits the default ROM table to be overridden (by implementing an external component connected to the external PPB to handle accesses to these addresses) without requiring every user of the processor to implement and connect a ROM table component.
- Because the CT model does not provide a DAP port or halting debug capability, the `dbgen` signal is ignored.

Implementation of ITM in M-class models

This model has a parameter that enables partial support for Instrumentation Trace Macrocell (ITM). In hardware or RTL, trace data from ITM is sent in packets to the trace block serially using a single pin or wire. In the model, if it is enabled, the ITM trace data is output using an MTI trace source called `ITM`. The `ITM` trace source has an `ITM_PACKET_TYPE` field. The following table shows which packet types the model supports:

Table 3-175 ITM_PACKET_TYPE field values that the model supports

Field value	Description	Supported by model
`ITM_SYNC`	Synchronization packet	Not supported.
`ITM_P_OVERFLOW`	Protocol: Overflow packet	Not supported.
`ITM_P_LOCAL_TIMESTAMP`	Protocol: Local timestamp packets	Not supported.
`ITM_P_GLOBAL_TIMESTAMP`	Protocol: Global timestamp packets	Not supported.
`ITM_P_EXTEN`	Protocol: Extension packet	Not supported.
`ITM_S_INSTRUMENTATION`	Source: Instrumentation packet	Supported.
`ITM_S_DWT_EVENT_COUNTER`	Hardware source: Event counter wrapping	Not supported.
`ITM_S_DWT_EXCEPTION`	Hardware source: Exception tracing	Supported.
`ITM_S_DWT_PC_SAMPLING`	Hardware source: PC sampling	Not supported.
`ITM_S_DWT_DATA_PC_TRACE`	Hardware source: DWT Data trace PC value	Supported.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-175 ITM_PACKET_TYPE field values that the model supports (continued)

Field value	Description	Supported by model
ITM_S_DWT_DATA_ADDRESS_TRACE	Hardware source: DWT Data trace address value	Supported.
ITM_S_DWT_DATA_DATA_TRACE	Hardware source: DWT Data trace DATA value	Supported.

Table 3-176 Ports

Name	Protocol	Type	Description
ahb_ap	PVBus	Slave	Debug AHB - core bus slave driven by the DAP.
auxfault	*Value* on page 2-85	Slave	This is wired to the Auxiliary Fault Status Register.
bigend	*Signal* on page 2-85	Slave	Configure big endian data format.
clk_in	ClockSignal	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the core executes instructions.
currpri	*Value* on page 2-85	Master	Current execution priority.
dbgen	*Signal* on page 2-85	Slave	Disallow (DAP) debugger access.
edbgrq	*Signal* on page 2-85	Slave	External debug request.
event	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE and corresponds to the RTL TXEV and RXEV signals.
fpudisable	*Signal* on page 2-85	Slave	Configure core with no FPU on reset.
fpxxc	*Value* on page 2-85	Master	Send the value of the FPXXC cumulative exception flags.
intisr[240]	*Signal* on page 2-85	Slave	This signal array delivers signals to the NVIC.
intnmi	*Signal* on page 2-85	Slave	Configure non maskable interrupt.
lockup	*Signal* on page 2-85	Master	Asserted when the processor is in lockup state.
mpudisable	*Signal* on page 2-85	Slave	Configure core with no MPU on reset.
poreset	*Signal* on page 2-85	Slave	Raising this signal will do a power-on reset of the core.
pv_ppbus_m	PVBus	Master	The core will generate External Private Peripheral Bus requests on this port.
pvbus_m	PVBus	Master	The core will generate bus requests on this port.
sleepdeep	*Signal* on page 2-85	Master	Asserted when the processor is in deep sleep.
sleeping	*Signal* on page 2-85	Master	Asserted when the processor is in sleep.
stcalib	*Value* on page 2-85	Slave	This is the calibration value for the SysTick timer.
stclk	ClockSignal	Slave	This is the reference clock for the SysTick timer.
sysreset	*Signal* on page 2-85	Slave	Raising this signal will put the core into reset mode (but does not reset the debug logic).
sysresetreq	*Signal* on page 2-85	Master	Asserted to indicate that a reset is required.
ticks	*InstructionCount* on page 2-84	Master	Port allowing the number of instructions since startup to be read from the CPU.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-177 Parameters for ARM_Cortex-M4

Name	Type	Default value	Description
BB_PRESENT	bool	0x1	Enable bitbanding
BIGENDINIT	bool	0x0	Initialize processor to big endian mode
DBGLVL	int	0x3	0: No debug; 1: Minimal debug; 2: Full debug without DWT address matching; 3: Full debug support with, DWT can compare data as well as address
LVL_WIDTH	int	0x3	Number of bits of interrupt priority
NUM_IRQ	int	0x10	Number of user interrupts
NUM_MPU_REGION	int	0x8	Number of MPU regions
TRACE_LVL	bool	0x1	Level of instrumentation trace supported. false : No ITM trace included, true: ITM trace included
WIC	bool	0x1	Include support for WIC-mode deep sleep
cpi_div	int	0x1	divider for calculating CPI (Cycles Per Instruction)
cpi_mul	int	0x1	multiplier for calculating CPI (Cycles Per Instruction)
master_id	int	0x0	Master ID presented in bus transactions
min_sync_level	int	0x0	force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)
scheduler_mode	int	0x0	Control the interleaving of instructions in this processor (0=default long quantum, 1=low latency mode, short quantum and signal checking, 2=lock-breaking mode, long quantum with additional context switches near load-exclusive instructions, 3=ISSCompare)
semihosting-Thumb_SVC	int	0xab	T32 SVC number for semihosting
semihosting-cmd_line	string	""	Command line available to semihosting SVC calls
semihosting-cwd	string	""	Base directory for semihosting file access.
semihosting-enable	bool	0x1	Enable semihosting SVC traps. Applications that do not use semihosting must set this parameter to false.
semihosting-heap_base	int	0x0	Virtual address of heap base
semihosting-heap_limit	int	0x10700000	Virtual address of top of heap
semihosting-prefix	bool	0x0	Prefix semihosting output with target instance name
semihosting-stack_base	int	0x10700000	Virtual address of base of descending stack
semihosting-stack_limit	int	0x10800000	Virtual address of stack limit
vfp-present	bool	0x1	Set whether the model has VFP support

3.4.35 ARMCortexM7CT

ARMCortexM7CT CPU component. This model is written in C++ and models version r0p2 of the RTL.

ARMCortexM7CT contains the following CADI targets:

- ARM_Cortex-M7

ARMCortexM7CT contains the following MTI components:

- *4.37 ARM_Cortex-M7 - trace* on page 4-2047
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.108 PVCache - trace* on page 4-2761

Implementation of ITM in M-class models

This model has a parameter that enables partial support for Instrumentation Trace Macrocell (ITM). In hardware or RTL, trace data from ITM is sent in packets to the trace block serially using a single pin or wire. In the model, if it is enabled, the ITM trace data is output using an MTI trace source called `ITM`. The `ITM` trace source has an `ITM_PACKET_TYPE` field. The following table shows which packet types the model supports:

Table 3-178 ITM_PACKET_TYPE field values that the model supports

Field value	Description	Supported by model
`ITM_SYNC`	Synchronization packet	Not supported.
`ITM_P_OVERFLOW`	Protocol: Overflow packet	Not supported.
`ITM_P_LOCAL_TIMESTAMP`	Protocol: Local timestamp packets	Not supported.
`ITM_P_GLOBAL_TIMESTAMP`	Protocol: Global timestamp packets	Not supported.
`ITM_P_EXTEN`	Protocol: Extension packet	Not supported.
`ITM_S_INSTRUMENTATION`	Source: Instrumentation packet	Supported.
`ITM_S_DWT_EVENT_COUNTER`	Hardware source: Event counter wrapping	Not supported.
`ITM_S_DWT_EXCEPTION`	Hardware source: Exception tracing	Supported.
`ITM_S_DWT_PC_SAMPLING`	Hardware source: PC sampling	Not supported.
`ITM_S_DWT_DATA_PC_TRACE`	Hardware source: DWT Data trace PC value	Supported.
`ITM_S_DWT_DATA_ADDRESS_TRACE`	Hardware source: DWT Data trace address value	Supported.
`ITM_S_DWT_DATA_DATA_TRACE`	Hardware source: DWT Data trace DATA value	Supported.

Differences between the model and the RTL

- TCMs are modeled internally and the model does not support external TCMs or the ports associated with them.
- ECC support is hardware-specific so is not modeled.

Table 3-179 Ports

Name	Protocol	Type	Description
`ahbd`	`PVBus`	Slave	Debug AHB - core bus slave driven by the DAP.
`ahbp_m`	`PVBus`	Master	The core will generate Vendor System data accesses on this port.
`ahbs`	`PVBus`	Slave	External master (e.g. DMA) can write TCMs (whether or not enabled in xTCMCR).
`auxfault`	*`Value` on page 2-85*	Slave	This is wired to the Auxiliary Fault Status Register.
`bigend`	*`Signal` on page 2-85*	Slave	Configure big endian data format.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-179 Ports (continued)

Name	Protocol	Type	Description
clk_in	ClockSignal	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the core executes instructions.
cpuwait	*Signal* on page 2-85	Slave	When this signal is HIGH out of reset, it forces the processor into a quiescent state that delays its boot-up sequence and instruction execution until this signal is driven LOW
currpri	*Value* on page 2-85	Master	Current execution priority.
dap_s	PVBus	Slave	Debug Access Port (DAP).
dbgen	*Signal* on page 2-85	Slave	Invasive debug enable.
dbgrestart	*Signal* on page 2-85	Slave	External debug request.
dbgrestarted	*Signal* on page 2-85	Master	External debug request.
edbgrq	*Signal* on page 2-85	Slave	External debug request.
event	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE and corresponds to the RTL TXEV and RXEV signals.
fpudisable	*Signal* on page 2-85	Slave	Configure core with no FPU on reset.
fpxxc	*Value* on page 2-85	Master	Port which sends the value of the FPXXC cumulative exception flags.
halted	*Signal* on page 2-85	Master	External debug request.
initvtor	*Value* on page 2-85	Slave	Initial value of the Vector Table Offset Register (VTOR)
intisr[240]	*Signal* on page 2-85	Slave	This signal array delivers signals to the NVIC.
intnmi	*Signal* on page 2-85	Slave	Configure non maskable interrupt.
lockup	*Signal* on page 2-85	Master	Asserted when the processor is in lockup state.
mpudisable	*Signal* on page 2-85	Slave	Configure core with no MPU on reset.
niden	*Signal* on page 2-85	Slave	Non-invasive debug enable.
poreset	*Signal* on page 2-85	Slave	Raising this signal will do a power-on reset of the core.
pv_ppbus_m	PVBus	Master	The core will generate External Private Peripheral Bus requests on this port.
pvbus_m	PVBus	Master	The core will generate bus requests on this port.
sleepdeep	*Signal* on page 2-85	Master	Asserted when the processor is in deep sleep.
sleeping	*Signal* on page 2-85	Master	Asserted when the processor is in sleep.
stcalib	*Value* on page 2-85	Slave	This is the calibration value for the SysTick timer.
stclk	ClockSignal	Slave	This is the reference clock for the SysTick timer.
sysreset	*Signal* on page 2-85	Slave	Raising this signal will put the core into reset mode (but does not reset the debug logic).
sysresetreq	*Signal* on page 2-85	Master	Asserted to indicate that a reset is required.
ticks	*InstructionCount* on page 2-84	Master	Port allowing the number of instructions since startup to be read from the CPU.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-180 Parameters for ARM_Cortex-M7

Name	Type	Default value	Description
BIGENDINIT	bool	0x0	Initialize processor to big endian mode
DBGLVL	int	0x1	0: 2 DWT, 4 FPB; 1: 4 DWT, 8 FPB comparators
DP_FLOAT	bool	0x1	Support 8-byte floats
INITVTOR	int	0x0	vector-table offset at reset
LVL_WIDTH	int	0x3	Number of bits of interrupt priority
NUM_IRQ	int	0x20	Number of user interrupts
NUM_MPU_REGION	int	0x10	Number of MPU regions
TRC	bool	0x1	Level of instrumentation trace supported. false : No ITM trace included, true: ITM trace included
WIC	bool	0x1	Include support for WIC-mode deep sleep
cpi_div	int	0x1	divider for calculating CPI (Cycles Per Instruction)
cpi_mul	int	0x1	multiplier for calculating CPI (Cycles Per Instruction)
dcache-size	int	0x8000	L1 D-cache size in bytes
dcache-state_modelled	bool	0x0	Set whether D-cache has stateful implementation
dcache-ways	int	0x4	L1 D-cache ways (sets are implicit from size)
dtcm_enable	bool	0x0	Enable DTCM at reset
dtcm_size	int	0x100	DTCM size in KB
duplicate_CADI_TCM_writes	bool	0x0	CADI writes to TCMs are also sent to downstream memory at same addresses (for validation platforms)
icache-size	int	0x8000	L1 I-cache size in bytes
icache-state_modelled	bool	0x0	Set whether I-cache has stateful implementation
icache-ways	int	0x2	L1 I-cache ways (sets are implicit from size)
itcm_enable	bool	0x0	Enable ITCM at reset
itcm_size	int	0x100	ITCM size in KB
master_id	int	0x0	Master ID presented in bus transactions
min_sync_level	int	0x0	force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)
scheduler_mode	int	0x0	Control the interleaving of instructions in this processor (0=default long quantum, 1=low latency mode, short quantum and signal checking, 2=lock-breaking mode, long quantum with additional context switches near load-exclusive instructions, 3=ISSCompare)
semihosting-Thumb_SVC	int	0xab	T32 SVC number for semihosting
semihosting-cmd_line	string	""	Command line available to semihosting SVC calls
semihosting-cwd	string	""	Base directory for semihosting file access.
semihosting-enable	bool	0x1	Enable semihosting SVC traps. Applications that do not use semihosting must set this parameter to false.
semihosting-heap_base	int	0x0	Virtual address of heap base

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-180 Parameters for ARM_Cortex-M7 (continued)

Name	Type	Default value	Description
semihosting-heap_limit	int	0x10700000	Virtual address of top of heap
semihosting-prefix	bool	0x0	Prefix semihosting output with target instance name
semihosting-stack_base	int	0x10700000	Virtual address of base of descending stack
semihosting-stack_limit	int	0x10800000	Virtual address of stack limit
vfp-present	bool	0x1	Set whether the model has VFP support

3.4.36 ARMCortexM23CT

ARMCortexM23CT CPU component. This model is written in C++.

ARMCortexM23CT contains the following CADI targets:

- ARM_Cortex-M23

ARMCortexM23CT contains the following MTI components:

- *4.31 ARM_Cortex-M23 - trace* on page 4-1870
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402
- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758

Differences between the CT model and RTL implementations

The model does not support MTB, ETM, or TPIU. MTB RAM is absent on the model.

Table 3-181 Ports

Name	Protocol	Type	Description
ahbd	PVBus	Slave	Debug AHB - core bus slave driven by the DAP.
bigend	*Signal* on page 2-85	Slave	Configure big endian data format.
clk_in	ClockSignal	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the core executes instructions.
cpuwait	*Signal* on page 2-85	Slave	Clear = Core goes through reset sequence as normal, Set = Core waits out of reset.
currpri	*Value* on page 2-85	Master	Current execution priority.
dap_s	PVBus	Slave	Debug Access Port (DAP).
dbgen	*Signal* on page 2-85	Slave	Invasive debug control signals. Debug enable, Set=enabled, Clear=disabled
dbgrestart	*Signal* on page 2-85	Slave	External request to leave debug state
dbgrestarted	*Signal* on page 2-85	Master	Acknowledge for DBGRESTART
edbgrq	*Signal* on page 2-85	Slave	External debug request.
event	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE and corresponds to the RTL TXEV and RXEV signals.
halted	*Signal* on page 2-85	Master	Core is in halt mode debug state
hreset	*Signal* on page 2-85	Slave	Raising this signal will put the core into reset mode (but does not reset the debug logic).

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-181 Ports (continued)

Name	Protocol	Type	Description
idau	PVBus	Master	The core will generate IDAU requests on this port.
idau_invalidate_region	*Value_64* on page 2-85	Slave	64 bit number to invalid IDAU memory ranage (start_address<<32|end_address)
initvtor	*Value* on page 2-85	Slave	Reset configuration port - Secure Vector table offset (VTOR.TBLOFF[31:7]) out of reset This port remains functional no matter ARMv8-M Security Extensions are included or not When ARMv8-M Security Extensions are not included, all exceptions will use NS vector base address given by this port.
initvtorns	*Value* on page 2-85	Slave	Reset configuration port - Non-Secure Vector table offset (VTOR.TBLOFF[31:7]) out of reset It becomes functional when ARMv8-M Security Extensions are included When ARMv8-M Security Extensions are not included, this port will be ignored.
io_port_in	PVBus	Slave	I/O port pair. See the documentation for the io_port_out port.
io_port_out	PVBus	Master	I/O port pair. Used if IOP is true. Transactions from io_port_out which do not "match" should be returned via io_port_in. For performance reasons, the I/O port interface is not modelled directly. Instead, a simple PVBus gasket is inserted at the point in the memory system where the I/O port would be. In hardware, a device would be attached to the port which would tell the CPU whether it would like to intercept each transaction, given its address. This can be modelled in a performant manner by connecting a PVBusMapper-based device to io_port_out which intercepts transactions of interest and passes other transactions back to the CPU via io_port_in. Your I/O port device model is also responsible for aborting transactions which would be aborted on hardware (e.g. exclusives) if necessary.
irq[240]	*Signal* on page 2-85	Slave	This signal array delivers signals to the NVIC.
lockup	*Signal* on page 2-85	Master	Asserted when the processor is in lockup state.
niden	*Signal* on page 2-85	Slave	Non-invasive debug enable, Set=enabled, Clear=disabled
nmi	*Signal* on page 2-85	Slave	Configure non maskable interrupt.
poreset	*Signal* on page 2-85	Slave	Raising this signal will do a power-on reset of the core.
pv_ppbus_m	PVBus	Master	The core will generate External Private Peripheral Bus requests on this port.
pvbus_m	PVBus	Master	The core will generate bus requests on this port.
sleepdeep	*Signal* on page 2-85	Master	Asserted when the processor is in deep sleep.
sleeping	*Signal* on page 2-85	Master	Asserted when the processor is in sleep.
spiden	*Signal* on page 2-85	Slave	Secure Debug enable , Set=enabled, Clear=disabled
spniden	*Signal* on page 2-85	Slave	Secure Non-invasive debug enable, Set=enabled, Clear=disabled

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-181 Ports (continued)

Name	Protocol	Type	Description
stcalib	*Value* on page 2-85	Slave	This is the calibration value for the Secure (or only, when ARMv8-M Security Extensions are not included) SysTick timer.
stcalibns	*Value* on page 2-85	Slave	This is the calibration value for the Non-Secure SysTick timer. When ARMv8-M Security Extensions are not included, this port will be ignored.
stclk	ClockSignal	Slave	This is the reference clock for the SysTick timer.
sysresetreq	*Signal* on page 2-85	Master	Asserted to indicate that a reset is required.
ticks	*InstructionCount* on page 2-84	Master	Port allowing the number of instructions since startup to be read from the CPU.

Table 3-182 Parameters for ARM_Cortex-M23

Name	Type	Default value	Description
BE	bool	0x0	Initialize processor to big endian mode
BKPT	int	0x4	Number of breakpoint unit comparators implemented
DBG	bool	0x1	Set whether debug extensions are implemented
INITVTOR	int	0x0	Secure vector-table offset at reset
INITVTORNS	int	0x0	Non-Secure vector-table offset at reset
IOP	bool	0x0	Send all d-side transactions to the port, io_port_out. Transactions which do not match should be returned to the port, io_port_in
IRQDIS0	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+0]
IRQDIS1	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+32]
IRQDIS2	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+64]
IRQDIS3	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+96]
IRQDIS4	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+128]
IRQDIS5	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+160]
IRQDIS6	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+192]
IRQDIS7	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+224]
MPU_NS	int	0x8	Number of regions in the Non-Secure MPU. If Security Extentions are absent, this is the total number of MPU regions
MPU_S	int	0x8	Number of regions in the Secure MPU. If Security Extentions are absent, this is ignored
NUMIRQ	int	0x10	Number of user interrupts
SAU	int	0x4	Number of SAU regions (0 => no SAU)
SAU_CTRL.ALLNS	bool	0x0	At reset, the SAU treats entire memory space as NS when the SAU is disabled if this is set
SAU_CTRL.ENABLE	bool	0x0	Enable SAU at reset
SAU_REGION0.BADDR	int	0x0	Base address of SAU region0 at reset

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-182 Parameters for ARM_Cortex-M23 (continued)

Name	Type	Default value	Description
SAU_REGION0.ENABLE	bool	0x0	Enable SAU region0 at reset
SAU_REGION0.LADDR	int	0x0	Limit address of SAU region0 at reset
SAU_REGION0.NSC	bool	0x0	Set NSC for SAU region0 at reset
SAU_REGION1.BADDR	int	0x0	Base address of SAU region1 at reset
SAU_REGION1.ENABLE	bool	0x0	Enable SAU region1 at reset
SAU_REGION1.LADDR	int	0x0	Limit address of SAU region1 at reset
SAU_REGION1.NSC	bool	0x0	Set NSC for SAU region1 at reset
SAU_REGION2.BADDR	int	0x0	Base address of SAU region2 at reset
SAU_REGION2.ENABLE	bool	0x0	Enable SAU region2 at reset
SAU_REGION2.LADDR	int	0x0	Limit address of SAU region2 at reset
SAU_REGION2.NSC	bool	0x0	Set NSC for SAU region2 at reset
SAU_REGION3.BADDR	int	0x0	Base address of SAU region3 at reset
SAU_REGION3.ENABLE	bool	0x0	Enable SAU region3 at reset
SAU_REGION3.LADDR	int	0x0	Limit address of SAU region3 at reset
SAU_REGION3.NSC	bool	0x0	Set NSC for SAU region3 at reset
SAU_REGION4.BADDR	int	0x0	Base address of SAU region4 at reset
SAU_REGION4.ENABLE	bool	0x0	Enable SAU region4 at reset
SAU_REGION4.LADDR	int	0x0	Limit address of SAU region4 at reset
SAU_REGION4.NSC	bool	0x0	Set NSC for SAU region4 at reset
SAU_REGION5.BADDR	int	0x0	Base address of SAU region5 at reset
SAU_REGION5.ENABLE	bool	0x0	Enable SAU region5 at reset
SAU_REGION5.LADDR	int	0x0	Limit address of SAU region5 at reset
SAU_REGION5.NSC	bool	0x0	Set NSC for SAU region5 at reset
SAU_REGION6.BADDR	int	0x0	Base address of SAU region6 at reset
SAU_REGION6.ENABLE	bool	0x0	Enable SAU region6 at reset
SAU_REGION6.LADDR	int	0x0	Limit address of SAU region6 at reset
SAU_REGION6.NSC	bool	0x0	Set NSC for SAU region6 at reset
SAU_REGION7.BADDR	int	0x0	Base address of SAU region7 at reset
SAU_REGION7.ENABLE	bool	0x0	Enable SAU region7 at reset
SAU_REGION7.LADDR	int	0x0	Limit address of SAU region7 at reset
SAU_REGION7.NSC	bool	0x0	Set NSC for SAU region7 at reset
SECEXT	bool	0x1	Whether the ARMv8-M Security Extensions are included
SYST	int	0x2	Include SysTick timer functionality (0=Absent, 1=Secure only, 2=Secure and NS)
VTOR	bool	0x1	Include Vector Table Offset Register

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-182 Parameters for ARM_Cortex-M23 (continued)

Name	Type	Default value	Description
WIC	bool	0x1	Include support for WIC-mode deep sleep
WPT	int	0x4	Number of watchpoint unit comparators implemented
cpi_div	int	0x1	divider for calculating CPI (Cycles Per Instruction)
cpi_mul	int	0x1	multiplier for calculating CPI (Cycles Per Instruction)
ignore-SCR.SLEEPONEXIT	bool	0x0	Never sleep on exit from handler to thread mode.
master_id	int	0x0	Master ID presented in bus transactions
min_sync_level	int	0x0	force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)
scheduler_mode	int	0x0	Control the interleaving of instructions in this processor (0=default long quantum, 1=low latency mode, short quantum and signal checking, 2=lock-breaking mode, long quantum with additional context switches near load-exclusive instructions, 3=ISSCompare)
semihosting-Thumb_SVC	int	0xab	T32 SVC number for semihosting
semihosting-cmd_line	string	""	Command line available to semihosting SVC calls
semihosting-cwd	string	""	Base directory for semihosting file access.
semihosting-enable	bool	0x1	Enable semihosting SVC traps. Applications that do not use semihosting must set this parameter to false.
semihosting-heap_base	int	0x0	Virtual address of heap base
semihosting-heap_limit	int	0x10700000	Virtual address of top of heap
semihosting-prefix	bool	0x0	Prefix semihosting output with target instance name
semihosting-stack_base	int	0x10700000	Virtual address of base of descending stack
semihosting-stack_limit	int	0x10800000	Virtual address of stack limit

3.4.37 ARMCortexM33CT

ARMCortexM33CT CPU component. This model is written in C++.

ARMCortexM33CT contains the following CADI targets:

- ARM_Cortex-M33

ARMCortexM33CT contains the following MTI components:

- *4.33 ARM_Cortex-M33 - trace* on page 4-1921
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402
- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758

About ARMCortexM33CT

The model supports Custom Datapath Extension (CDE). For more information, see *5.5 CDE* on page 5-2899.

Differences between the model and the RTL

The model does not support the following:

- ETM, MTB, CTI, or TPIU. MTB RAM is absent on the model.
- The power control (Q-Channel) interface.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Implementation of ITM in M-class models

This model has a parameter that enables partial support for Instrumentation Trace Macrocell (ITM). In hardware or RTL, trace data from ITM is sent in packets to the trace block serially using a single pin or wire. In the model, if it is enabled, the ITM trace data is output using an MTI trace source called `ITM`. The `ITM` trace source has an `ITM_PACKET_TYPE` field. The following table shows which packet types the model supports:

Table 3-183 ITM_PACKET_TYPE field values that the model supports

Field value	Description	Supported by model
ITM_SYNC	Synchronization packet	Not supported.
ITM_P_OVERFLOW	Protocol: Overflow packet	Not supported.
ITM_P_LOCAL_TIMESTAMP	Protocol: Local timestamp packets	Not supported.
ITM_P_GLOBAL_TIMESTAMP	Protocol: Global timestamp packets	Not supported.
ITM_P_EXTEN	Protocol: Extension packet	Not supported.
ITM_S_INSTRUMENTATION	Source: Instrumentation packet	Supported.
ITM_S_DWT_EVENT_COUNTER	Hardware source: Event counter wrapping	Not supported.
ITM_S_DWT_EXCEPTION	Hardware source: Exception tracing	Supported.
ITM_S_DWT_PC_SAMPLING	Hardware source: PC sampling	Not supported.
ITM_S_DWT_DATA_PC_TRACE	Hardware source: DWT Data trace PC value	Supported.
ITM_S_DWT_DATA_ADDRESS_TRACE	Hardware source: DWT Data trace address value	Supported.
ITM_S_DWT_DATA_DATA_TRACE	Hardware source: DWT Data trace DATA value	Supported.

Table 3-184 Ports

Name	Protocol	Type	Description
ahbd	PVBus	Slave	Debug AHB - core bus slave driven by the DAP.
auxfault	*Value* on page 2-85	Slave	This is wired to the Auxiliary Fault Status Register.
bigend	*Signal* on page 2-85	Slave	Configure big endian data format.
clk_in	ClockSignal	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the core executes instructions. Referred to in some documents as 'CLKIN'.
coproc_bus	*CoprocBusProtocol* on page 2-81	Slave	Co-Processor Interface
cpuwait	*Signal* on page 2-85	Slave	Stall the CPU out of reset
currns	*Signal* on page 2-85	Master	Current Security state of the processor
currpri	*Value* on page 2-85	Master	Current execution priority.
dbgen	*Signal* on page 2-85	Slave	Invasive debug enable
dbgrestart	*Signal* on page 2-85	Slave	Request for synchronised exit from halt mode

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-184 Ports (continued)

Name	Protocol	Type	Description
dbgrestarted	*Signal* on page 2-85	Master	Handshakes with DBGRESTART
edbgrq	*Signal* on page 2-85	Slave	External request to enter halt mode
event	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE and corresponds to the RTL TXEV and RXEV signals.
fpxxc	*Value* on page 2-85	Master	Port which sends the value of the FPXXC cumulative exception flags.
halted	*Signal* on page 2-85	Master	Indicates that the processor is in halt mode
idau	PVBus	Master	The core will generate IDAU requests on this port.
idau_invalidate_region	*Value_64* on page 2-85	Slave	64 bit number to invalid IDAU memory ranage (start_address<<32|end_address)
initnsvtor	*Value* on page 2-85	Slave	Reset configuration port - Non-Secure Vector table offset (VTOR.TBLOFF[31:7]) out of reset This port remains functional no matter ARMv8-M Security Extensions are included or not When ARMv8-M Security Extensions are not included, all exceptions will use NS vector base address given by this port.
initsvtor	*Value* on page 2-85	Slave	Reset configuration port - Secure Vector table offset (VTOR.TBLOFF[31:7]) out of reset It becomes functional when ARMv8-M Security Extensions are included When ARMv8-M Security Extensions are not included, this port will be ignored.
intnum	*Value* on page 2-85	Master	Exception number of the current execution context (from IPSR[8:0]) When the processor is in Thread mode, INTNUM is 0 When the processor is in Handler mode, INTNUM is the exception number of the currently-executing exception.
irq[480]	*Signal* on page 2-85	Slave	This signal array delivers signals to the NVIC.
locknsmpu	*Signal* on page 2-85	Slave	Disable writes to the Non-Secure MPU_*_NS registers
locknsvtor	*Signal* on page 2-85	Slave	Cortex-M33-specific LOCKNSVTOR, LOCKSVTAIRCR, LOCKSMPU, LOCKNSMPU, LOCKSAU. Disable writes to VTOR_NS
locksau	*Signal* on page 2-85	Slave	Disable writes to the SAU_* registers
locksmpu	*Signal* on page 2-85	Slave	Disable writes to the Secure MPU_* registers
locksvtaircr	*Signal* on page 2-85	Slave	Disable writes to VTOR_S, AIRCR.PRIS, AIRCR.BFHFNMINS
lockup	*Signal* on page 2-85	Master	Asserted when the processor is in lockup state.
niden	*Signal* on page 2-85	Slave	Non-invasive debug enable
nmi	*Signal* on page 2-85	Slave	Configure non maskable interrupt.
poreset	*Signal* on page 2-85	Slave	Raising this signal will do a power-on reset of the core.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-184 Ports (continued)

Name	Protocol	Type	Description
pv_ppbus_m	PVBus	Master	The core will generate External Private Peripheral Bus requests on this port.
pvbus_m	PVBus	Master	The core will generate bus requests on this port.
sleepdeep	*Signal* on page 2-85	Master	Asserted when the processor is in deep sleep.
sleeping	*Signal* on page 2-85	Master	Asserted when the processor is in sleep.
spiden	*Signal* on page 2-85	Slave	Secure invasive debug enable
spniden	*Signal* on page 2-85	Slave	Secure non-invasive debug enable
stcalib[2]	*Value* on page 2-85	Slave	This is the calibration value for the SysTick timer.
stclk	ClockSignal	Slave	This is the reference clock for the SysTick timer.
sysreset	*Signal* on page 2-85	Slave	Raising this signal will put the core into reset mode (but does not reset the debug logic).
sysresetreq	*Signal* on page 2-85	Master	Asserted to indicate that a reset is required.
ticks	*InstructionCount* on page 2-84	Master	Port allowing the number of instructions since startup to be read from the CPU.
wicenack	*Signal* on page 2-85	Master	Acknowledge signal for WICENREQ
wicenreq	*Signal* on page 2-85	Slave	Request for deep sleep to be WIC-based deep sleep.
wicsense[483]	*Signal* on page 2-85	Master	Indicates which input events can cause the WIC to generate the WAKEUP signal.

Table 3-185 Parameters for ARM_Cortex-M33

Name	Type	Default value	Description
BIGENDINIT	bool	0x0	Initialize processor to big endian mode
CDEMAPPEDONCP	bool	0x1	Specifies whether the instruction for a given Coprocessor is redirected to the CDE module
CFGNOCDECP	int	0x0	Bit N means external coprocessor N (CP7:CP0) disable for CDE coprocessor
CPIF	bool	0x1	Specifies whether the external coprocessor interface is included
CPNSPRESENT	int	0xff	Bit N means external coprocessor N (CP7:CP0) is accessible in Non-Secure state
CPSPRESENT	int	0xff	Bit N means external coprocessor N (CP7:CP0) is accessible in Secure state
DBGLVL	int	0x2	0: Minimal debug; 1: 2 Watchpoints, 4 Breakpoint comparators; 2: 4 Watchpoints, 8 Breakpoint comparators
DSP	bool	0x1	Set whether the model has the DSP extension
FPU	bool	0x1	Set whether the model has VFP support
INITNSVTOR	int	0x0	Non-Secure vector-table offset at reset
INITSVTOR	int	0x0	Secure vector-table offset at reset

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-185 Parameters for ARM_Cortex-M33 (continued)

Name	Type	Default value	Description
IRQDIS0	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+0]
IRQDIS1	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+32]
IRQDIS10	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+320]
IRQDIS11	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+352]
IRQDIS12	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+384]
IRQDIS13	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+416]
IRQDIS14	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+448]
IRQDIS2	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+64]
IRQDIS3	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+96]
IRQDIS4	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+128]
IRQDIS5	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+160]
IRQDIS6	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+192]
IRQDIS7	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+224]
IRQDIS8	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+256]
IRQDIS9	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+288]
IRQLVL	int	0x3	Number of bits of interrupt priority
ITM	bool	0x1	Level of instrumentation trace supported. false : No ITM trace included, true: ITM trace included
LOCK_NS_MPU	bool	0x0	Lock down of Non-Secure MPU registers write
LOCK_SAU	bool	0x0	Lock down of SAU registers write
LOCK_S_MPU	bool	0x0	Lock down of Secure MPU registers write
MPU_NS	int	0x8	Number of regions in the Non-Secure MPU. If Security Extentions are absent, this is the total number of MPU regions
MPU_S	int	0x8	Number of regions in the Secure MPU. If Security Extentions are absent, this is ignored
NUMIRQ	int	0x20	Number of user interrupts

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-185 Parameters for ARM_Cortex-M33 (continued)

Name	Type	Default value	Description
SAU	int	0x4	Number of SAU regions (0 => no SAU)
SAU_CTRL.ALLNS	bool	0x0	At reset, the SAU treats entire memory space as NS when the SAU is disabled if this is set
SAU_CTRL.ENABLE	bool	0x0	Enable SAU at reset
SAU_REGION0.BADDR	int	0x0	Base address of SAU region0 at reset
SAU_REGION0.ENABLE	bool	0x0	Enable SAU region0 at reset
SAU_REGION0.LADDR	int	0x0	Limit address of SAU region0 at reset
SAU_REGION0.NSC	bool	0x0	Set NSC for SAU region0 at reset
SAU_REGION1.BADDR	int	0x0	Base address of SAU region1 at reset
SAU_REGION1.ENABLE	bool	0x0	Enable SAU region1 at reset
SAU_REGION1.LADDR	int	0x0	Limit address of SAU region1 at reset
SAU_REGION1.NSC	bool	0x0	Set NSC for SAU region1 at reset
SAU_REGION2.BADDR	int	0x0	Base address of SAU region2 at reset
SAU_REGION2.ENABLE	bool	0x0	Enable SAU region2 at reset
SAU_REGION2.LADDR	int	0x0	Limit address of SAU region2 at reset
SAU_REGION2.NSC	bool	0x0	Set NSC for SAU region2 at reset
SAU_REGION3.BADDR	int	0x0	Base address of SAU region3 at reset
SAU_REGION3.ENABLE	bool	0x0	Enable SAU region3 at reset
SAU_REGION3.LADDR	int	0x0	Limit address of SAU region3 at reset
SAU_REGION3.NSC	bool	0x0	Set NSC for SAU region3 at reset
SAU_REGION4.BADDR	int	0x0	Base address of SAU region4 at reset
SAU_REGION4.ENABLE	bool	0x0	Enable SAU region4 at reset
SAU_REGION4.LADDR	int	0x0	Limit address of SAU region4 at reset
SAU_REGION4.NSC	bool	0x0	Set NSC for SAU region4 at reset
SAU_REGION5.BADDR	int	0x0	Base address of SAU region5 at reset
SAU_REGION5.ENABLE	bool	0x0	Enable SAU region5 at reset
SAU_REGION5.LADDR	int	0x0	Limit address of SAU region5 at reset
SAU_REGION5.NSC	bool	0x0	Set NSC for SAU region5 at reset
SAU_REGION6.BADDR	int	0x0	Base address of SAU region6 at reset
SAU_REGION6.ENABLE	bool	0x0	Enable SAU region6 at reset
SAU_REGION6.LADDR	int	0x0	Limit address of SAU region6 at reset
SAU_REGION6.NSC	bool	0x0	Set NSC for SAU region6 at reset
SAU_REGION7.BADDR	int	0x0	Base address of SAU region7 at reset
SAU_REGION7.ENABLE	bool	0x0	Enable SAU region7 at reset
SAU_REGION7.LADDR	int	0x0	Limit address of SAU region7 at reset

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-185 Parameters for ARM_Cortex-M33 (continued)

Name	Type	Default value	Description
SAU_REGION7.NSC	bool	0x0	Set NSC for SAU region7 at reset
SECEXT	bool	0x1	Whether the ARMv8-M Security Extensions are included
WIC	bool	0x1	Include support for WIC-mode deep sleep
WICLINES	int	0x23	Number of lines supported by the WIC interface
cde_impl_name	string	""	Name of the CDE implementation for this core (implementation contributed by MTI plugin).
cpi_div	int	0x1	divider for calculating CPI (Cycles Per Instruction)
cpi_mul	int	0x1	multiplier for calculating CPI (Cycles Per Instruction)
has_cde	bool	0x0	Enables Custom Datapath Extensions
ignore-SCR.SLEEPONEXIT	bool	0x0	Never sleep on exit from handler to thread mode.
master_id	int	0x0	Master ID presented in bus transactions
min_sync_level	int	0x0	force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)
scheduler_mode	int	0x0	Control the interleaving of instructions in this processor (0=default long quantum, 1=low latency mode, short quantum and signal checking, 2=lock-breaking mode, long quantum with additional context switches near load-exclusive instructions, 3=ISSCompare)
semihosting-Thumb_SVC	int	0xab	T32 SVC number for semihosting
semihosting-cmd_line	string	""	Command line available to semihosting SVC calls
semihosting-cwd	string	""	Base directory for semihosting file access.
semihosting-enable	bool	0x1	Enable semihosting SVC traps. Applications that do not use semihosting must set this parameter to false.
semihosting-heap_base	int	0x0	Virtual address of heap base
semihosting-heap_limit	int	0x10700000	Virtual address of top of heap
semihosting-prefix	bool	0x0	Prefix semihosting output with target instance name
semihosting-stack_base	int	0x10700000	Virtual address of base of descending stack
semihosting-stack_limit	int	0x10800000	Virtual address of stack limit
vfp-enable_at_reset	bool	0x0	Enable VFP in CPACR, CPPWR, NSACR at reset. Warning: Arm recommends going through the implementation's suggested VFP power-up sequence!
write_unknown_regs_at_exception	bool	0x0	DEPRECATED and non-functional in device models.

3.4.38 ARMCortexM35PCT

ARMCortexM35PCT CPU component. This model is written in C++.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ARMCortexM35PCT contains the following CADI targets:

- ARM_Cortex-M35P

ARMCortexM35PCT contains the following MTI components:

- *4.34 ARM_Cortex-M35P - trace* on page 4-1954
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402
- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758

Differences between the model and the RTL

- The model does not support the following:
  - ETM, MTB, CTI, or TPIU. MTB RAM is absent on the model.
  - Caches.
  - The co-processor interface.
  - The power control (Q-Channel) interface.
- The model does not implement any physical security features.
- Bits[3:0] of the Anti-tampering Features Control Register are supported for read/write. No functionality is implemented.
- Read/write access to the Anti-tampering Features Control Register is supported using SECKEY. No functionality is implemented.

Implementation of ITM in M-class models

This model has a parameter that enables partial support for Instrumentation Trace Macrocell (ITM). In hardware or RTL, trace data from ITM is sent in packets to the trace block serially using a single pin or wire. In the model, if it is enabled, the ITM trace data is output using an MTI trace source called `ITM`. The `ITM` trace source has an `ITM_PACKET_TYPE` field. The following table shows which packet types the model supports:

Table 3-186 ITM_PACKET_TYPE field values that the model supports

Field value	Description	Supported by model
`ITM_SYNC`	Synchronization packet	Not supported.
`ITM_P_OVERFLOW`	Protocol: Overflow packet	Not supported.
`ITM_P_LOCAL_TIMESTAMP`	Protocol: Local timestamp packets	Not supported.
`ITM_P_GLOBAL_TIMESTAMP`	Protocol: Global timestamp packets	Not supported.
`ITM_P_EXTEN`	Protocol: Extension packet	Not supported.
`ITM_S_INSTRUMENTATION`	Source: Instrumentation packet	Supported.
`ITM_S_DWT_EVENT_COUNTER`	Hardware source: Event counter wrapping	Not supported.
`ITM_S_DWT_EXCEPTION`	Hardware source: Exception tracing	Supported.
`ITM_S_DWT_PC_SAMPLING`	Hardware source: PC sampling	Not supported.
`ITM_S_DWT_DATA_PC_TRACE`	Hardware source: DWT Data trace PC value	Supported.
`ITM_S_DWT_DATA_ADDRESS_TRACE`	Hardware source: DWT Data trace address value	Supported.
`ITM_S_DWT_DATA_DATA_TRACE`	Hardware source: DWT Data trace DATA value	Supported.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-187 Ports

Name	Protocol	Type	Description
LOCKATFCR	*Signal* on page 2-85	Slave	Port Lock ATFCR register
auxfault	*Value* on page 2-85	Slave	This is wired to the Auxiliary Fault Status Register.
bigend	*Signal* on page 2-85	Slave	Configure big endian data format.
clk_in	ClockSignal	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the core executes instructions. Referred to in some documents as 'CLKIN'.
cpuwait	*Signal* on page 2-85	Slave	Stall the CPU out of reset
currns	*Signal* on page 2-85	Master	Current Security state of the processor
currpri	*Value* on page 2-85	Master	Current execution priority.
dbgen	*Signal* on page 2-85	Slave	Invasive debug enable
dbgrestart	*Signal* on page 2-85	Slave	Request for synchronised exit from halt mode
dbgrestarted	*Signal* on page 2-85	Master	Handshakes with DBGRESTART
edbgrq	*Signal* on page 2-85	Slave	External request to enter halt mode
event	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE and corresponds to the RTL TXEV and RXEV signals.
fpxxc	*Value* on page 2-85	Master	Port which sends the value of the FPXXC cumulative exception flags.
halted	*Signal* on page 2-85	Master	Indicates that the processor is in halt mode
idau	PVBus	Master	The core will generate IDAU requests on this port.
idau_invalidate_region	*Value_64* on page 2-85	Slave	64 bit number to invalid IDAU memory ranage (start_address<<32|end_address)
initnsvtor	*Value* on page 2-85	Slave	Reset configuration port - Non-Secure Vector table offset (VTOR.TBLOFF[31:7]) out of reset This port remains functional no matter ARMv8-M Security Extensions are included or not When ARMv8-M Security Extensions are not included, all exceptions will use NS vector base address given by this port.
initsvtor	*Value* on page 2-85	Slave	Reset configuration port - Secure Vector table offset (VTOR.TBLOFF[31:7]) out of reset It becomes functional when ARMv8-M Security Extensions are included When ARMv8-M Security Extensions are not included, this port will be ignored.
intnum	*Value* on page 2-85	Master	Exception number of the current execution context (from IPSR[8:0]) When the processor is in Thread mode, INTNUM is 0 When the processor is in Handler mode, INTNUM is the exception number of the currently-executing exception.
irq[480]	*Signal* on page 2-85	Slave	This signal array delivers signals to the NVIC.
locknsmpu	*Signal* on page 2-85	Slave	Disable writes to the Non-Secure MPU_*_NS registers

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-187 Ports (continued)

Name	Protocol	Type	Description
locknsvtor	*Signal* on page 2-85	Slave	Disable writes to VTOR_NS
locksau	*Signal* on page 2-85	Slave	Disable writes to the SAU_* registers
locksmpu	*Signal* on page 2-85	Slave	Disable writes to the Secure MPU_* registers
locksvtaircr	*Signal* on page 2-85	Slave	Disable writes to VTOR_S, AIRCR.PRIS, AIRCR.BFHFNMINS
lockup	*Signal* on page 2-85	Master	Asserted when the processor is in lockup state.
niden	*Signal* on page 2-85	Slave	Non-invasive debug enable
nmi	*Signal* on page 2-85	Slave	Configure non maskable interrupt.
poreset	*Signal* on page 2-85	Slave	Raising this signal will do a power-on reset of the core.
pv_ppbus_m	PVBus	Master	The core will generate External Private Peripheral Bus requests on this port.
pvbus_m	PVBus	Master	The core will generate bus requests on this port.
sleepdeep	*Signal* on page 2-85	Master	Asserted when the processor is in deep sleep.
sleeping	*Signal* on page 2-85	Master	Asserted when the processor is in sleep.
spiden	*Signal* on page 2-85	Slave	Secure invasive debug enable
spniden	*Signal* on page 2-85	Slave	Secure non-invasive debug enable
stcalib[2]	*Value* on page 2-85	Slave	This is the calibration value for the SysTick timer.
stclk	ClockSignal	Slave	This is the reference clock for the SysTick timer.
sysreset	*Signal* on page 2-85	Slave	Raising this signal will put the core into reset mode (but does not reset the debug logic).
sysresetreq	*Signal* on page 2-85	Master	Asserted to indicate that a reset is required.
ticks	*InstructionCount* on page 2-84	Master	Port allowing the number of instructions since startup to be read from the CPU.
wicenack	*Signal* on page 2-85	Master	Acknowledge signal for WICENREQ
wicenreq	*Signal* on page 2-85	Slave	Request for deep sleep to be WIC-based deep sleep.
wicsense[483]	*Signal* on page 2-85	Master	Indicates which input events can cause the WIC to generate the WAKEUP signal.

Table 3-188 Parameters for ARM_Cortex-M35P

Name	Type	Default value	Description
ATFINITEN	bool	0x0	ATFCR is enabled when the core goes out of reset
BIGENDINIT	bool	0x0	Initialize processor to big endian mode
CPIF	bool	0x1	Specifies whether the external coprocessor interface is included
CPNSPRESENT	int	0xff	Bit N means external coprocessor N (CP7:CP0) is accessible in Non-Secure state
CPSPRESENT	int	0xff	Bit N means external coprocessor N (CP7:CP0) is accessible in Secure state

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-188 Parameters for ARM_Cortex-M35P (continued)

Name	Type	Default value	Description
DBGLVL	int	0x2	0: Minimal debug; 1: 2 Watchpoints, 4 Breakpoint comparators; 2: 4 Watchpoints, 8 Breakpoint comparators
DSP	bool	0x1	Set whether the model has the DSP extension
FPU	bool	0x1	Set whether the model has VFP support
INITNSVTOR	int	0x0	Non-Secure vector-table offset at reset
INITSVTOR	int	0x0	Secure vector-table offset at reset
IRQDIS0	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+0]
IRQDIS1	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+32]
IRQDIS10	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+320]
IRQDIS11	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+352]
IRQDIS12	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+384]
IRQDIS13	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+416]
IRQDIS14	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+448]
IRQDIS2	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+64]
IRQDIS3	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+96]
IRQDIS4	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+128]
IRQDIS5	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+160]
IRQDIS6	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+192]
IRQDIS7	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+224]
IRQDIS8	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+256]
IRQDIS9	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+288]
IRQLVL	int	0x3	Number of bits of interrupt priority
ITM	bool	0x1	Level of instrumentation trace supported. false : No ITM trace included, true: ITM trace included
LOCK_NS_MPU	bool	0x0	Lock down of Non-Secure MPU registers write
LOCK_SAU	bool	0x0	Lock down of SAU registers write
LOCK_S_MPU	bool	0x0	Lock down of Secure MPU registers write
MPU_NS	int	0x8	Number of regions in the Non-Secure MPU. If Security Extentions are absent, this is the total number of MPU regions
MPU_S	int	0x8	Number of regions in the Secure MPU. If Security Extentions are absent, this is ignored
NUMIRQ	int	0x20	Number of user interrupts
SAU	int	0x4	Number of SAU regions (0 => no SAU)
SAU_CTRL.ALLNS	bool	0x0	At reset, the SAU treats entire memory space as NS when the SAU is disabled if this is set
SAU_CTRL.ENABLE	bool	0x0	Enable SAU at reset

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-188 Parameters for ARM_Cortex-M35P (continued)

Name	Type	Default value	Description
SAU_REGION0.BADDR	int	0x0	Base address of SAU region0 at reset
SAU_REGION0.ENABLE	bool	0x0	Enable SAU region0 at reset
SAU_REGION0.LADDR	int	0x0	Limit address of SAU region0 at reset
SAU_REGION0.NSC	bool	0x0	Set NSC for SAU region0 at reset
SAU_REGION1.BADDR	int	0x0	Base address of SAU region1 at reset
SAU_REGION1.ENABLE	bool	0x0	Enable SAU region1 at reset
SAU_REGION1.LADDR	int	0x0	Limit address of SAU region1 at reset
SAU_REGION1.NSC	bool	0x0	Set NSC for SAU region1 at reset
SAU_REGION2.BADDR	int	0x0	Base address of SAU region2 at reset
SAU_REGION2.ENABLE	bool	0x0	Enable SAU region2 at reset
SAU_REGION2.LADDR	int	0x0	Limit address of SAU region2 at reset
SAU_REGION2.NSC	bool	0x0	Set NSC for SAU region2 at reset
SAU_REGION3.BADDR	int	0x0	Base address of SAU region3 at reset
SAU_REGION3.ENABLE	bool	0x0	Enable SAU region3 at reset
SAU_REGION3.LADDR	int	0x0	Limit address of SAU region3 at reset
SAU_REGION3.NSC	bool	0x0	Set NSC for SAU region3 at reset
SAU_REGION4.BADDR	int	0x0	Base address of SAU region4 at reset
SAU_REGION4.ENABLE	bool	0x0	Enable SAU region4 at reset
SAU_REGION4.LADDR	int	0x0	Limit address of SAU region4 at reset
SAU_REGION4.NSC	bool	0x0	Set NSC for SAU region4 at reset
SAU_REGION5.BADDR	int	0x0	Base address of SAU region5 at reset
SAU_REGION5.ENABLE	bool	0x0	Enable SAU region5 at reset
SAU_REGION5.LADDR	int	0x0	Limit address of SAU region5 at reset
SAU_REGION5.NSC	bool	0x0	Set NSC for SAU region5 at reset
SAU_REGION6.BADDR	int	0x0	Base address of SAU region6 at reset
SAU_REGION6.ENABLE	bool	0x0	Enable SAU region6 at reset
SAU_REGION6.LADDR	int	0x0	Limit address of SAU region6 at reset
SAU_REGION6.NSC	bool	0x0	Set NSC for SAU region6 at reset
SAU_REGION7.BADDR	int	0x0	Base address of SAU region7 at reset
SAU_REGION7.ENABLE	bool	0x0	Enable SAU region7 at reset
SAU_REGION7.LADDR	int	0x0	Limit address of SAU region7 at reset
SAU_REGION7.NSC	bool	0x0	Set NSC for SAU region7 at reset
SECEXT	bool	0x1	Whether the ARMv8-M Security Extensions are included
WIC	bool	0x1	Include support for WIC-mode deep sleep
WICLINES	int	0x23	Number of lines supported by the WIC interface

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-188 Parameters for ARM_Cortex-M35P (continued)

Name	Type	Default value	Description
cde_impl_name	string	""	Name of the CDE implementation for this core (implementation contributed by MTI plugin).
cpi_div	int	0x1	divider for calculating CPI (Cycles Per Instruction)
cpi_mul	int	0x1	multiplier for calculating CPI (Cycles Per Instruction)
has_cde	bool	0x0	Enables Custom Datapath Extensions
ignore-SCR.SLEEPONEXIT	bool	0x0	Never sleep on exit from handler to thread mode.
master_id	int	0x0	Master ID presented in bus transactions
min_sync_level	int	0x0	force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)
scheduler_mode	int	0x0	Control the interleaving of instructions in this processor (0=default long quantum, 1=low latency mode, short quantum and signal checking, 2=lock-breaking mode, long quantum with additional context switches near load-exclusive instructions, 3=ISSCompare)
semihosting-Thumb_SVC	int	0xab	T32 SVC number for semihosting
semihosting-cmd_line	string	""	Command line available to semihosting SVC calls
semihosting-cwd	string	""	Base directory for semihosting file access.
semihosting-enable	bool	0x1	Enable semihosting SVC traps. Applications that do not use semihosting must set this parameter to false.
semihosting-heap_base	int	0x0	Virtual address of heap base
semihosting-heap_limit	int	0x10700000	Virtual address of top of heap
semihosting-prefix	bool	0x0	Prefix semihosting output with target instance name
semihosting-stack_base	int	0x10700000	Virtual address of base of descending stack
semihosting-stack_limit	int	0x10800000	Virtual address of stack limit
vfp-enable_at_reset	bool	0x0	Enable VFP in CPACR, CPPWR, NSACR at reset. Warning: Arm recommends going through the implementation's suggested VFP power-up sequence!

3.4.39 ARMCortexM55CT

CortexM55CT CPU component. This model is written in C++.

ARMCortexM55CT contains the following CADI targets:

- ARM_Cortex-M55

ARMCortexM55CT contains the following MTI components:

- *4.36 ARM_Cortex-M55 - trace* on page 4-2009
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402
- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.108 PVCache - trace* on page 4-2761

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

About ARMCortexM55CT

The model supports Custom Datapath Extension (CDE). For more information, see *5.5 CDE on page 5-2899*.

The model does not support the following functionality:

- Cross Trigger Interface (CTI).
- Programmable MBIST controller (PMC-100).
- Error Correcting Code (ECC).
- Q-Channel.

The following interfaces and registers are not modeled:

- ITM and ETM trace and trace synchronization and trigger interface signals.
- Dual-core lock-step operation.
- Interrupt latencies.
- Memory System Control Register (MSCR).
- Prefetcher Control Register (PFCR).
- Direct cache access registers.

Differences between the FVP_MPS2 model and the MPS2 specification

- The model does not have the random number generator or unique ID/secure storage mentioned in the MPS2 specification because the programmer's view of these devices is yet to be defined.
- The Ethernet controller in the model is a LAN91C111. The MPS2 documents, including those for pre-v8-M cores, specify a LAN9220.
- MTB, ETM, and TPIU are not supported. MTB RAM is absent.
- In the Memory Gating Unit, the model provides a configurable block size. For performance reasons, the minimum block size in the model is 4096 bytes. Hardware and later models might allow smaller block sizes. Software should always use the `BLK_CFG` register to determine block size.
- As in previous MPS2 Fast Models, some of the peripherals have minimal implementations:
  - The Audio controller is RAZ/WI.
  - Only the touchscreen functionality of the STMPE811 touchscreen controller is implemented.
  - A subset of the Ampire LCD module's graphics modes are supported.
- The usual general Fast Models restrictions apply.

Implementation of ITM in M-class models

This model has a parameter that enables partial support for Instrumentation Trace Macrocell (ITM). In hardware or RTL, trace data from ITM is sent in packets to the trace block serially using a single pin or wire. In the model, if it is enabled, the ITM trace data is output using an MTI trace source called `ITM`. The `ITM` trace source has an `ITM_PACKET_TYPE` field. The following table shows which packet types the model supports:

Table 3-189 ITM_PACKET_TYPE field values that the model supports

Field value	Description	Supported by model
`ITM_SYNC`	Synchronization packet	Not supported.
`ITM_P_OVERFLOW`	Protocol: Overflow packet	Not supported.
`ITM_P_LOCAL_TIMESTAMP`	Protocol: Local timestamp packets	Not supported.
`ITM_P_GLOBAL_TIMESTAMP`	Protocol: Global timestamp packets	Not supported.
`ITM_P_EXTEN`	Protocol: Extension packet	Not supported.
`ITM_S_INSTRUMENTATION`	Source: Instrumentation packet	Supported.
`ITM_S_DWT_EVENT_COUNTER`	Hardware source: Event counter wrapping	Not supported.
`ITM_S_DWT_EXCEPTION`	Hardware source: Exception tracing	Supported.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-189 ITM_PACKET_TYPE field values that the model supports (continued)

Field value	Description	Supported by model
ITM_S_DWT_PC_SAMPLING	Hardware source: PC sampling	Not supported.
ITM_S_DWT_DATA_PC_TRACE	Hardware source: DWT Data trace PC value	Supported.
ITM_S_DWT_DATA_ADDRESS_TRACE	Hardware source: DWT Data trace address value	Supported.
ITM_S_DWT_DATA_DATA_TRACE	Hardware source: DWT Data trace DATA value	Supported.

Table 3-190 Ports

Name	Protocol	Type	Description
ahbd	PVBus	Slave	-
ahbp_m	PVBus	Master	The core will generate Vendor System data accesses on this port.
auxfault	*Value* on page 2-85	Slave	This is wired to the Auxiliary Fault Status Register.
bigend	*Signal* on page 2-85	Slave	Configure big endian data format.
clk_in	ClockSignal	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the core executes instructions. Referred to in some documents as 'CLKIN'.
coproc_bus	*CoprocBusProtocol* on page 2-81	Slave	Co-Processor Interface
cpuwait	*Signal* on page 2-85	Slave	Stall the CPU out of reset
currns	*Signal* on page 2-85	Master	Current Security state of the processor
currpri	*Value* on page 2-85	Master	Current execution priority.
dbgen	*Signal* on page 2-85	Slave	Invasive debug enable
dbgrestart	*Signal* on page 2-85	Slave	Request for synchronised exit from halt mode
dbgrestarted	*Signal* on page 2-85	Master	Handshakes with DBGRESTART
edbgrq	*Signal* on page 2-85	Slave	External request to enter halt mode
event	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE and corresponds to the RTL TXEV and RXEV signals.
fpxxc	*Value* on page 2-85	Master	Port which sends the value of the FPXXC cumulative exception flags.
halted	*Signal* on page 2-85	Master	Indicates that the processor is in halt mode
idau	PVBus	Master	The core will generate IDAU requests on this port.
idau_invalidate_region	*Value_64* on page 2-85	Slave	64 bit number to invalid IDAU memory ranage (start_address<<32|end_address)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-190 Ports (continued)

Name	Protocol	Type	Description
initnsvtor	*Value* on page 2-85	Slave	Reset configuration port - Non-Secure Vector table offset (VTOR.TBLOFF[31:7]) out of reset This port remains functional no matter ARMv8-M Security Extensions are included or not When ARMv8-M Security Extensions are not included, all exceptions will use NS vector base address given by this port.
initpahben	*Signal* on page 2-85	Slave	Enable P-AHB on the next reset
initsvtor	*Value* on page 2-85	Slave	Reset configuration port - Secure Vector table offset (VTOR.TBLOFF[31:7]) out of reset It becomes functional when ARMv8-M Security Extensions are included When ARMv8-M Security Extensions are not included, this port will be ignored.
inittcmen[2]	*Signal* on page 2-85	Slave	Reset configuration port - TCM enable initialisation out of reset Bit[0] HIGH = ITCM is enabled Bit[1] HIGH = DTCM is enabled
intnum	*Value* on page 2-85	Master	Exception number of the current execution context (from IPSR[8:0]) When the processor is in Thread mode, INTNUM is 0 When the processor is in Handler mode, INTNUM is the exception number of the currently-executing exception.
irq[480]	*Signal* on page 2-85	Slave	This signal array delivers signals to the NVIC.
lockdcaic	*Signal* on page 2-85	Slave	-
locknsmpu	*Signal* on page 2-85	Slave	Disable writes to the Non-Secure MPU_*_NS registers
locknsvtor	*Signal* on page 2-85	Slave	Cortex-M55-specific LOCKNSVTOR, LOCKSVTAIRCR, LOCKSMPU, LOCKNSMPU, LOCKSAU. Disable writes to VTOR_NS
lockpahb	*Signal* on page 2-85	Slave	P-AHB related ports Disable writes to PAHBCR
locksau	*Signal* on page 2-85	Slave	Disable writes to the SAU_* registers
locksmpu	*Signal* on page 2-85	Slave	Disable writes to the Secure MPU_* registers
locksvtaircr	*Signal* on page 2-85	Slave	Disable writes to VTOR_S, AIRCR.PRIS, AIRCR.BFHFNMINS
lockup	*Signal* on page 2-85	Master	Asserted when the processor is in lockup state.
niden	*Signal* on page 2-85	Slave	Non-invasive debug enable
nmi	*Signal* on page 2-85	Slave	Configure non maskable interrupt.
poreset	*Signal* on page 2-85	Slave	Raising this signal will do a power-on reset of the core.
prequest	*PChannel* on page 2-80	Slave	Low Power Interface
pv_ppbus_m	PVBus	Master	The core will generate External Private Peripheral Bus requests on this port.
pvbus_m	PVBus	Master	The core will generate bus requests on this port.
qrequest	*PChannel* on page 2-80	Slave	-

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-190 Ports (continued)

Name	Protocol	Type	Description
sleepdeep	*Signal* on page 2-85	Master	Asserted when the processor is in deep sleep.
sleeping	*Signal* on page 2-85	Master	Asserted when the processor is in sleep.
spiden	*Signal* on page 2-85	Slave	Secure invasive debug enable
spniden	*Signal* on page 2-85	Slave	Secure non-invasive debug enable
stcalib[2]	*Value* on page 2-85	Slave	This is the calibration value for the SysTick timer.
stclk	ClockSignal	Slave	This is the reference clock for the SysTick timer.
sysreset	*Signal* on page 2-85	Slave	Raising this signal will put the core into reset mode (but does not reset the debug logic).
sysresetreq	*Signal* on page 2-85	Master	Asserted to indicate that a reset is required.
ticks	*InstructionCount* on page 2-84	Master	Port allowing the number of instructions since startup to be read from the CPU.
wicenack	*Signal* on page 2-85	Master	Acknowledge signal for WICENREQ
wicenreq	*Signal* on page 2-85	Slave	Request for deep sleep to be WIC-based deep sleep.
wicsense[483]	*Signal* on page 2-85	Master	Indicates which input events can cause the WIC to generate the WAKEUP signal.

Table 3-191 Parameters for ARM_Cortex-M55

Name	Type	Default value	Description
BEATS_PER_TICK	int	0x2	Number of beats from each in-flight vector instruction executed in 1 tick (1,2 or 4).
BF_is_nop	bool	0x0	BF instruction executes as NOP, even if we have LO_BRANCH_INFO.
CDEMAPPEDONCP	bool	0x1	Specifies whether the instruction for a given Coprocessor is redirected to the CDE module
CFGBIGEND	bool	0x0	Initialize processor to big endian mode
CFGDTCMSZ	int	0x9	Size of the data TCM. 0=No DTCM implemented. Otherwise=Size of DTCM=pow(2, CFGDTCMSZ - 1) KB. Minimum size is 4KB
CFGITCMSZ	int	0x9	Size of the instruction TCM. 0=No ITCM implemented. Otherwise=Size of ITCM=pow(2, CFGITCMSZ - 1) KB. Minimum size is 4KB
CFGMEMALIAS	int	0x0	Memory address alias bit for the ITCM, DTCM and P-AHB regions. 0=No alias, 1=Alias bit 24, 2=Alias bit 25, 4=Alias bit 26, 8=Alias bit 27, 16=Alias bit 28
CFGNOCDECP	int	0x0	Bit N means external coprocessor N (CP7:CP0) disable for CDE coprocessor

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-191 Parameters for ARM_Cortex-M55 (continued)

Name	Type	Default value	Description
CFGPAHBSZ	int	0x0	Size of the P-AHB peripheral port memory region. 0=P-AHB disabled, 1=64MB, 2=128MB, 3=256MB, 4=512MB
CPIF	bool	0x1	Specifies whether the external coprocessor interface is included
CPNSPRESENT	int	0xff	Bit N means external coprocessor N (CP7:CP0) is accessible in Non-Secure state
CPSPRESENT	int	0xff	Bit N means external coprocessor N (CP7:CP0) is accessible in Secure state
DBGLVL	int	0x2	0: Minimal debug; 1: 2 Watchpoints, 4 Breakpoint comparators; 2: 4 Watchpoints, 8 Breakpoint comparators; 3: 8 Watchpoints, 8 Breakpoint comparators
DCACHESZ	int	0xf	Whether the D-cache is included and, if included, the size of it. Bit 0: 0=No D-cache included, 1=D-cache included. Bits [4:1]: 0x0=4KB D-cache, 0x1=8KB D-cache, 0x3=16KB D-cache, 0x7=32KB D-cache, 0x15=64KB D-cache
DTGU	bool	0x0	DTCM Security Gate Unit included
DTGUBLKSZ	int	0x3	DTCM gate unit block size. Size=pow(2, DTGUBLKSZ + 5) bytes
DTGUMAXBLKS	int	0x0	Maxiumum number of DTCM gate unit blocks. Number of blocks=pow(2, DTGUMAXBLKS)
ECOREVNUM	int	0x0	ECO Revision number
ERRDEVID.NUM	int	0x38	RAS: Number of implemented error record indexes, 0 to 56.
ETM	bool	0x1	Support for ETM trace. false : No ETM trace included, true: ETM trace included
FPU	bool	0x1	Set whether the model has VFP support
ICACHESZ	int	0xf	Whether the I-cache is included and, if included, the size of it. Bit 0: 0=No I-cache included, 1=I-cache included. Bits [4:1]: 0x0=4KB I-cache, 0x1=8KB I-cache, 0x3=16KB I-cache, 0x7=32KB I-cache, 0x15=64KB I-cache

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-191 Parameters for ARM_Cortex-M55 (continued)

Name	Type	Default value	Description
ID_ISAR0.CmpBranch	int	0x3	Support for Compare and Branch instructions. 1 = Supports CBNZ and CBZ instructions; 3 = Supports non-predicated low overhead looping (WLS, DLS, LE, and LC) and branch future (BF, BFX, BFL, BFLX, and BFCSEL) instructions.
INITNSVTOR	int	0x0	Non-Secure vector-table offset at reset
INITPAHBEN	bool	0x0	The P-AHB enable state at reset
INITSVTOR	int	0x0	Secure vector-table offset at reset
IRQDIS0	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+0]
IRQDIS1	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+32]
IRQDIS10	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+320]
IRQDIS11	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+352]
IRQDIS12	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+384]
IRQDIS13	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+416]
IRQDIS14	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+448]
IRQDIS2	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+64]
IRQDIS3	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+96]
IRQDIS4	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+128]
IRQDIS5	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+160]
IRQDIS6	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+192]
IRQDIS7	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+224]
IRQDIS8	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+256]
IRQDIS9	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n+288]
IRQLVL	int	0x3	Number of bits of interrupt priority
ITGU	bool	0x0	ITCM Security Gate Unit included

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-191 Parameters for ARM_Cortex-M55 (continued)

Name	Type	Default value	Description
ITGUBLKSZ	int	0x3	ITCM gate unit block size. Size=pow(2, ITGUBLKSZ + 5) bytes
ITGUMAXBLKS	int	0x0	Maxiumum number of ITCM gate unit blocks. Number of blocks=pow(2, ITGUMAXBLKS)
ITM	bool	0x1	Level of instrumentation trace supported. false : No ITM trace included, true: ITM trace included
IWIC	bool	0x1	Include support for Internal Wake-up Interrupt Controller
LOCKDTGU	bool	0x0	Lock down of Data TGU registers write
LOCKITGU	bool	0x0	Lock down of Instruction TGU registers write
LOCKTCM	bool	0x0	Lock down of TCM registers write
LOCK_NS_MPU	bool	0x0	Lock down of Non-Secure MPU registers write
LOCK_SAU	bool	0x0	Lock down of SAU registers write
LOCK_S_MPU	bool	0x0	Lock down of Secure MPU registers write
MPU_NS	int	0x8	Number of regions in the Non-Secure MPU. If Security Extentions are absent, this is the total number of MPU regions
MPU_S	int	0x8	Number of regions in the Secure MPU. If Security Extentions are absent, this is ignored
MVE	int	0x1	Set whether the model has MVE support. If FPU = 0: 0=MVE not included, 1=Integer subset of MVE included. If FPU = 1: 0=MVE not included, 1=Integer subset of MVE included, 2=Integer and half and single precision floating point MVE included
NUMIRQ	int	0x20	Number of user interrupts
SAU	int	0x4	Number of SAU regions (0 => no SAU)
SECEXT	bool	0x1	Whether the ARMv8-M Security Extensions are included
WICLINES	int	0x23	Number of lines supported by the WIC interface
aircr_iesb_is_writable	bool	0x1	IS the AIRCR.IESB bit [5] writable?
aircr_iesb_reset	bool	0x0	Set the AIRCR.IESB bit [5] after reset

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-191 Parameters for ARM_Cortex-M55 (continued)

Name	Type	Default value	Description
cde_impl_name	string	""	Name of the CDE implementation for this core (implementation contributed by MTI plugin).
cpi_div	int	0x1	divider for calculating CPI (Cycles Per Instruction)
cpi_mul	int	0x1	multiplier for calculating CPI (Cycles Per Instruction)
dcache-state_modelled	bool	0x0	Set whether D-cache has stateful implementation
dcache-ways	int	0x4	L1 D-cache ways (sets are implicit from size)
delay_faultmask_update	bool	0x0	Delay FAULTMASK update to context sync
delay_sysreg_update	bool	0x0	Delay some system register updates (e.g. SHCSR) to context sync
duplicate_CADI_TCM_writes	bool	0x0	CADI writes to TCMs are also sent to downstream memory at same addresses (for validation platforms)
ecc_on	bool	0x0	Enable Error Correcting Code
execute_via_archex	bool	0x1	Use ArchEx-generated code from V8_1_PACBTI_EAC_2_0_INTERNAL for execution
has_cde	bool	0x0	Enables Custom Datapath Extensions
has_pmu	bool	0x0	Availability of optional PMU.
has_unpriviledged_debug	bool	0x1	Unpriviledged Debug Extension supported for Mainline Extension
icache-state_modelled	bool	0x0	Set whether I-cache has stateful implementation
icache-ways	int	0x2	L1 I-cache ways (sets are implicit from size)
master_id	int	0x0	Master ID presented in bus transactions
min_sync_level	int	0x0	force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)
mve_unpred_config_json	string	'"[["CLEAR_IT","VPNOT_T1","InITBlock"],["OK","VMINNMV_f_T2","Rda==11x1"],["OK","VDDUP","curOffset MOD imm32 != 0"],["OK","VDDUP","bufSize MOD imm32 != 0"],["OK","VDDUP","curOffset >= bufSize"]]"'	A JSON array of arrays of the form [unpred_result, instr, cond] forunpredictable configruation for MVE. Either instr or cond may be omitted. The first match wins. Use 'list' to show the available options
num_pmu_counters	int	0x1f	Number of available PMU counters.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-191 Parameters for ARM_Cortex-M55 (continued)

Name	Type	Default value	Description
ras_ERRFR0	string	'"{"ED":0x1,"UE":0x1}"'	A JSON object or array of objects for each field of ERRFR. Records not described default to RAZ e.g. '{"ED":0x1,"UE":0x1}'.
ras_cei_pin	int	0x2	RAS: Critical error interrupt pin.
ras_cei_support	bool	0x1	RAS: Whether Critical Error Interrupt is supported
ras_eri_pin	int	0x1	RAS: Error recovery interrupt pin.
ras_eri_support	bool	0x1	RAS: Whether Error Recovery Interrupt is supported
ras_error_record	int	0xffffffffffffff	56 bit value that specifies which nodes out of 0-55 are implemented (ERRDEVID is derived from this parameter)
ras_fhi_pin	int	0x0	RAS: Fault handling interrupt pin.
ras_fhi_support	bool	0x1	RAS: Whether Fault Handling Interrupt is supported
scheduler_mode	int	0x0	Control the interleaving of instructions in this processor (0=default long quantum, 1=low latency mode, short quantum and signal checking, 2=lock-breaking mode, long quantum with additional context switches near load-exclusive instructions, 3=ISSCompare)
semihosting-Thumb_SVC	int	0xab	T32 SVC number for semihosting
semihosting-cmd_line	string	""	Command line available to semihosting SVC calls
semihosting-cwd	string	""	Base directory for semihosting file access.
semihosting-enable	bool	0x1	Enable semihosting SVC traps. Applications that do not use semihosting must set this parameter to false.
semihosting-heap_base	int	0x0	Virtual address of heap base
semihosting-heap_limit	int	0x10700000	Virtual address of top of heap
semihosting-prefix	bool	0x0	Prefix semihosting output with target instance name
semihosting-stack_base	int	0x10700000	Virtual address of base of descending stack
semihosting-stack_limit	int	0x10800000	Virtual address of stack limit
tcm_fill_pattern_1	int	0xdfdfdfcf	TCM Fill pattern 1
tcm_fill_pattern_2	int	0xcfdfdfdf	TCM Fill pattern 2

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-191 Parameters for ARM_Cortex-M55 (continued)

Name	Type	Default value	Description
`trace_style`	`int`	`0x0`	MVE instruction trace style: 0=Tarmac-like from instDB.json, 1=execute function+params (for debug), 2 = Rosetta. Add 16 for [**--] beat trace. Add 32 for tracing IMPLIED LOB instructions. Add 64 to change opcode of implied BF to 0xBF00
`unpred_config_json_file`	`string`	`""`	Path to the unpredictable configuration file in JSON format
`vfp-enable_at_reset`	`bool`	`0x0`	Enable VFP in CPACR, CPPWR, NSACR at reset. Warning: Arm recommends going through the implementation's suggested VFP power-up sequence!

3.4.40 ARMCortexR4CT

ARMCortexR4CT CPU component. This model is written in C++ and models version r1p2 of the RTL.

ARMCortexR4CT contains the following CADI targets:

- ARM_Cortex-R4
- PVCache

ARMCortexR4CT contains the following MTI components:

- *4.38 ARM_Cortex-R4 - trace* on page 4-2080
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402
- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.108 PVCache - trace* on page 4-2761

About ARMCortexR4CT

- The model implements the `cfgie` port, although it is optional in hardware.
- `pvbus_s` is the slave port to access the TCM RAM. Bits [3:0] of the user flags in the transaction are used to select the TCM:
  - 1 selects the ATCM.
  - 2 selects the BTCM.
  - Any other value is reserved.
- When you use the `semihosting-cmd_line` parameter to set the command line that is available to semihosting SVC calls, the value of `argv[0]` is set to the first command-line argument, not to the name of an image.

Differences between the model and the RTL

This component has the following differences from the corresponding revision of the RTL implementation:

- There is a single memory port combining instruction, data, DMA and peripheral access.
- ECC and parity schemes are not supported (although the registers might be present).
- The dual core redundancy configuration is not supported.
- TCMs are modeled internally and the model does not support external TCMs or the ports associated with them.
- The hardware refers to the TCMs as “A” and “B”. The model refers to these as “i” and “d”.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

- The RTL permits two data TCMs, B0 and B1, to be configured for extra bandwidth. These are not modeled.
- The `vfp-enable_at_reset` option is model-specific and has no hardware equivalent. Arm recommends that it is only used in test systems and tied off to `false` in production systems.

Table 3-192 Ports

Name	Protocol	Type	Description
cfgend0	*Signal* on page 2-85	Slave	Configure BE8 mode after a reset.
cfgie	*Signal* on page 2-85	Slave	Configure big endian instruction format after a reset.
cfgnmfi	*Signal* on page 2-85	Slave	Configure FIQs as non-maskable after a reset.
cfgte	*Signal* on page 2-85	Slave	Configure exceptions to be taken in thumb mode after a reset.
clk_in	ClockSignal	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the core executes instructions.
cpuhalt	*Signal* on page 2-85	Slave	Raising this signal will put the core into halt mode.
fiq	*Signal* on page 2-85	Slave	This signal drives the CPU's fast-interrupt handling.
initramd	*Signal* on page 2-85	Slave	Configure DTCM enabled after a reset.
initrami	*Signal* on page 2-85	Slave	Configure ITCM enabled after a reset.
irq	*Signal* on page 2-85	Slave	This signal drives the CPU's interrupt handling.
loczrama	*Signal* on page 2-85	Slave	Location of ATCM at reset.
pmuirq	*Signal* on page 2-85	Master	Interrupt signal from performance monitoring unit.
pvbus_m	PVBus	Master	The core will generate bus requests on this port.
pvbus_s	PVBus	Slave	Slave access to TCMs.
reset	*Signal* on page 2-85	Slave	Raising this signal will put the core into reset mode.
standbywfi	*Signal* on page 2-85	Master	Signal from the core that it is waiting in standby for an interrupt.
ticks	*InstructionCount* on page 2-84	Master	Port allowing the number of instructions since startup to be read from the CPU.
vic_ack	*Signal* on page 2-85	Master	Vic acknowledge port to primary VIC.
vic_addr	*Value* on page 2-85	Slave	Vic address port from primary VIC.
vinithi	*Signal* on page 2-85	Slave	Configure high vectors after a reset.

Table 3-193 Parameters for ARM_Cortex-R4

Name	Type	Default value	Description
CFGEND0	bool	0x0	Initialize to BE8 endianness
CFGIE	bool	0x0	Set the reset value of the instruction endian bit
CFGNMFI	bool	0x0	Enable nonmaskable FIQ interrupts on startup
CFGTE	bool	0x0	Initialize to take exceptions in T32 state. Model starts in T32 state
INITRAMD	bool	0x0	Set or reset the INITRAMD signal
INITRAMI	bool	0x0	Set or reset the INITRAMI signal
LOCZRAMI	bool	0x0	Set or reset the LOCZRAMI signal

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-193 Parameters for ARM_Cortex-R4 (continued)

Name	Type	Default value	Description
NUM_MPU_REGION	int	0x8	Number of MPU regions
VINITHI	bool	0x0	Initialize with high vectors enabled
cpi_div	int	0x1	Divider for calculating CPI (Cycles Per Instruction)
cpi_mul	int	0x1	Multiplier for calculating CPI (Cycles Per Instruction)
dcache-size	int	0x10000	Set D-cache size in bytes
dcache-state_modelled	bool	0x0	Set whether D-cache has stateful implementation
dtcm0_base	int	0x800000	Base address of DTCM at startup
dtcm0_size	int	0x8	Size of DTCM in KB
icache-size	int	0x10000	Set I-cache size in bytes
icache-state_modelled	bool	0x0	Set whether I-cache has stateful implementation
implements_vfp	bool	0x1	Set whether the model has been built with VFP support
itcm0_base	int	0x0	Base address of ITCM at startup
itcm0_size	int	0x8	Size of ITCM in KB
master_id	int	0x0	Master ID presented in bus transactions
min_sync_level	int	0x0	force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)
scheduler_mode	int	0x0	Control the interleaving of instructions in this processor. 0, default long quantum. 1, low latency mode, short quantum and signal checking. 2, lock-breaking mode, long quantum with additional context switches near load-exclusive instructions.
semihosting-ARM_HLT	int	0xf000	ARM HLT number for semihosting
semihosting-ARM_SVC	int	0x123456	ARM SVC number for semihosting
semihosting-Thumb_HLT	int	0x3c	Thumb HLT number for semihosting
semihosting-Thumb_SVC	int	0xab	Thumb SVC number for semihosting
semihosting-cmd_line	string	""	Command line available to semihosting SVC calls
semihosting-cwd	string	""	Base directory for semihosting file access.
semihosting-enable	bool	0x1	Enable semihosting SVC traps. Applications that do not use semihosting must set this parameter to false
semihosting-heap_base	int	0x0	Virtual address of heap base
semihosting-heap_limit	int	0xf000000	Virtual address of top of heap
semihosting-hlt-enable	bool	0x0	Enable semihosting HLT traps. Applications that use HLT semihosting must set this parameter to true and the semihosting-enable parameter to true
semihosting-prefix	bool	0x0	Prefix semihosting output with target instance name
semihosting-stack_base	int	0x10000000	Virtual address of base of descending stack
semihosting-stack_limit	int	0xf000000	Virtual address of stack limit
vfp-enable_at_reset	bool	0x0	Enable coprocessor access and VFP at reset

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

3.4.41 ARMCortexR5x1CT

ARMCortexR5x1CT CPU component. This model is written in C++ and models version r1p2 of the RTL.

ARMCortexR5x1CT contains the following CADI targets:

- ARM_Cortex-R5
- Cluster_ARM_Cortex-R5

ARMCortexR5x1CT contains the following MTI components:

- *4.39 ARM_Cortex-R5 - trace* on page 4-2103
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402
- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.108 PVCache - trace* on page 4-2761

About ARMCortexR5x1CT

- An ARMCortexR5x2CT component also exists.
- The per-core parameters are preceded by `cpu`*n*`.`, where *n* identifies the core (0 or 1).
- The allowed values for the `LOCK_STEP` parameter are:

  0 Disable. Set for two independent cores.

  1 Lock Step. Appears to the system as two cores but is internally modeled as a single core.

  3 Split Lock. Appears to the system as two cores but can be statically configured from reset either as two independent cores or two locked cores. For the model, these are equivalent to Disable and Lock Step, respectively, except for the value of build options registers. The model does not support dynamically splitting and locking the cluster.
- `pvbus_s` is the slave port to access the TCM RAM of CPU *n*. Bits [3:0] of the user flags in the transaction are used to select the TCM:
  - 1 selects the ATCM of CPU 0.
  - 2 selects the BTCM of CPU 0.
  - 3 selects the ATCM of CPU 1.
  - 4 selects the BTCM of CPU 1.
  - Any other value is reserved.

Differences between the model and the RTL

This component has the following differences from the corresponding revision of the RTL implementation:

- The RR bit in the SCTLR is ignored.
- The Low Latency Peripheral Port is not modeled.
- The model only has a single bus master port combining instruction, data, DMA and peripheral accesses. The CP15 control registers associated with peripheral buses preserve values but do not have any other effect.
- The model only supports static split lock and not dynamic split lock. Contact Arm for details.
- TCMs are modeled internally and the model does not support external TCMs or the ports associated with them.
- The model cannot experience an ECC error and does not support fault injection into the system, so Arm does not provide the ability to set error schemes for the caches or TCMs. Contact Arm if you require a particular value in the Build Options registers.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-194 Ports

Name	Protocol	Type	Description
acp_s	PVBus	Slave	ACP slave port.
cfgatcmsz[2]	*Value* on page 2-85	Slave	ATCM size.
cfgbtcmsz[2]	*Value* on page 2-85	Slave	BTCM Size.
cfgend[2]	*Signal* on page 2-85	Slave	This signal is for EE bit initialisation. This is CFGEE in RTL but cfgend here fastsim consistency reasons.
cfgnmfi[2]	*Signal* on page 2-85	Slave	Controls non-maskable Fast Interrupts.
clk_in	ClockSignal	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the core executes instructions.
cpuhalt[2]	*Signal* on page 2-85	Slave	Raising this signal will put the core into halt mode.
event[2]	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE.
fiq[2]	*Signal* on page 2-85	Slave	This signal drives the CPU's fast-interrupt handling.
groupid	*Value* on page 2-85	Slave	Group ID used for MPIDR.
initrama[2]	*Signal* on page 2-85	Slave	If ATCM is enabled at reset.
initramb[2]	*Signal* on page 2-85	Slave	If BTCM is enabled at reset.
irq[2]	*Signal* on page 2-85	Slave	This signal drives the CPU's interrupt handling.
loczrama[2]	*Signal* on page 2-85	Slave	Location of ATCM at reset.
pmuirq[2]	*Signal* on page 2-85	Master	Interrupt signal from performance monitoring unit.
pvbus_m	PVBus	Master	The core will generate bus requests on this port.
pvbus_s[1]	PVBus	Slave	tcm slave port.
reset[2]	*Signal* on page 2-85	Slave	Raising this signal will put the core into reset mode.
standbywfe[2]	*Signal* on page 2-85	Master	This signal indicate if a core is in wfe state RTL calls this WFEPIPESTOPPED.
standbywfi[2]	*Signal* on page 2-85	Master	This signal indicates if a core is in WFI state RTL uses WFIPIPESTOPPED.
teinit[2]	*Signal* on page 2-85	Slave	Default exception handling state.
ticks[2]	*InstructionCount* on page 2-84	Master	This port should be connected to one of the two ticks ports on a 'visualisation' component, in order to display a running instruction count.
vic_ack[2]	*Signal* on page 2-85	Master	Vic acknowledge port to primary VIC.
vic_addr[2]	*Value* on page 2-85	Slave	Vic address port from primary VIC.
vinithi[2]	*Signal* on page 2-85	Slave	This signal controls of the location of the exception vectors at reset.

Table 3-195 Parameters for Cluster_ARM_Cortex-R5

Name	Type	Default value	Description
GROUP_ID	int	0x0	Value read in GROUP ID register field, bits[15:8] of the MPIDR
INST_ENDIAN	bool	0x1	Controls whether the model supports the instruction endianness bit

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-195 Parameters for Cluster_ARM_Cortex-R5 (continued)

Name	Type	Default value	Description
LOCK_STEP	int	0x0	Affects dual-processor configurations only, and ignored by single-processor configurations
MICRO_SCU	bool	0x1	Controls whether the effects of the MicroSCU are modeled
NUM_BREAKPOINTS	int	0x3	Controls with how many breakpoint pairs the model has been configured. This only affects the build options registers, because debug is not modeled
NUM_MPU_REGION	int	0xc	Sets the number of MPU regions
NUM_WATCHPOINTS	int	0x2	Controls with how many watchpoint pairs the model has been configured. This only affects the build options registers, because debug is not modeled
SLSPLIT	bool	0x0	Sets whether the model starts in split mode or locked mode
cpi_div	int	0x1	Divider for calculating CPI (Cycles Per Instruction)
cpi_mul	int	0x1	Multiplier for calculating CPI (Cycles Per Instruction)
dcache-state_modelled	bool	0x0	Set whether D-cache has stateful implementation
icache-state_modelled	bool	0x0	Set whether I-cache has stateful implementation
scheduler_mode	int	0x0	Control the interleaving of instructions in this processor. 0, default long quantum. 1, low latency mode, short quantum and signal checking. 2, lock-breaking mode, long quantum with additional context switches near load-exclusive instructions.

Table 3-196 Parameters for ARM_Cortex-R5

Name	Type	Default value	Description
cpu0.CFGATCMSZ	int	0xe	Sets the size of the ATCM
cpu0.CFGBTCMSZ	int	0xe	Sets the size of the BTCM
cpu0.CFGEND	bool	0x0	Initialize to BE8 endianness
cpu0.CFGIE	bool	0x0	Set the reset value of the instruction endian bit
cpu0.CFGNMFI	bool	0x0	Enable nonmaskable FIQ interrupts on startup
cpu0.DP_FLOAT	bool	0x1	Sets whether double-precision instructions are available
cpu0.INITRAMA	bool	0x0	Initialize with TCMA enabled
cpu0.INITRAMB	bool	0x0	Initialize with TCMB enabled
cpu0.LOCZRAMA	bool	0x0	Initialize with LOCZRAMA set to 1
cpu0.TEINIT	bool	0x0	T32 exception enable. The default has exceptions including reset handled in A32 state
cpu0.VINITHI	bool	0x0	Initialize with high vectors enabled
cpu0.atcm_base	int	0x40000000	Model-specific. Sets the base address of the ATCM (forced to 0 if LOCZRAMA is 1)
cpu0.btcm_base	int	0x0	Model-specific. Sets the base address of the BTCM (forced to 0 if LOCZRAMA is 0)
cpu0.dcache-size	int	0x10000	Set D-cache size in bytes
cpu0.icache-size	int	0x10000	Set I-cache size in bytes

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-196 Parameters for ARM_Cortex-R5 (continued)

Name	Type	Default value	Description
cpu0.min_sync_level	int	0x0	Force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)
cpu0.semihosting-ARM_HLT	int	0xf000	ARM HLT number for semihosting
cpu0.semihosting-ARM_SVC	int	0x123456	ARM SVC number for semihosting
cpu0.semihosting-Thumb_HLT	int	0x3c	Thumb HLT number for semihosting
cpu0.semihosting-Thumb_SVC	int	0xab	Thumb SVC number for semihosting
cpu0.semihosting-cmd_line	string	""	Command line available to semihosting SVC calls
cpu0.semihosting-cwd	string	""	Base directory for semihosting file access.
cpu0.semihosting-enable	bool	0x1	Enable semihosting SVC traps. Applications that do not use semihosting must set this parameter to false
cpu0.semihosting-heap_base	int	0x0	Virtual address of heap base
cpu0.semihosting-heap_limit	int	0xf000000	Virtual address of top of heap
cpu0.semihosting-hlt-enable	bool	0x0	Enable semihosting HLT traps. Applications that use HLT semihosting must set this parameter to true and the semihosting-enable parameter to true
cpu0.semihosting-prefix	bool	0x0	Prefix semihosting output with target instance name
cpu0.semihosting-stack_base	int	0x10000000	Virtual address of base of descending stack
cpu0.semihosting-stack_limit	int	0xf000000	Virtual address of stack limit
cpu0.vfp-enable_at_reset	bool	0x0	Enable coprocessor access and VFP at reset
cpu0.vfp-present	bool	0x1	Set whether model has VFP support

3.4.42 ARMCortexR7x1CT

ARMCortexR7x1CT CPU component. This model is written in C++ and models version r0p0 of the RTL.

ARMCortexR7x1CT contains the following CADI targets:

- ARM_Cortex-R7
- Cluster_ARM_Cortex-R7
- PVCache

ARMCortexR7x1CT contains the following MTI components:

- *4.40 ARM_Cortex-R7 - trace* on page 4-2126
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402
- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.108 PVCache - trace* on page 4-2761

About ARMCortexR7x1CT

- An ARMCortexR7x2CT component also exists.
- The per-core parameters are preceded by `cpu`*n*`.`, where *n* identifies the core (0 or 1).
- `pvbus_s` is the slave port to access the TCM RAM of CPU *n*. Bits [3:0] of the user flags in the transaction are used to select the TCM:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

  - — 0 selects the ITCM of CPU 0.
  - — 1 selects the DTCM of CPU 0.
  - — 2 selects the ITCM of CPU 1.
  - — 3 selects the DTCM of CPU 1.
  - — Any other value is reserved.
- When you use the `semihosting-cmd_line` parameter to set the command line that is available to semihosting SVC calls, the value of `argv[0]` is set to the first command-line argument, not to the name of an image.
- The `vfp-enable_at_reset` option is model-specific and has no hardware equivalent. Arm recommends that it is only used in test systems and tied off to `false` in production systems.

Differences between the model and the RTL

This component has the following differences from the corresponding revision of the RTL implementation:

- This component does not implement address filtering within the SCU. The enable bit for this feature is ignored.
- The GIC does not respect the CFGSDISABLE signal. This leads to some registers being accessible when they must not be.
- The SCU enable bit is ignored. The SCU is always enabled.
- The SCU ignores the invalidate all register.
- The Broadcast TLB or cache operations in this model do not cause other cores in the cluster that are asleep because of WFI to wake up.
- The RR bit in the SCTLR is ignored.
- The Power Control Register in the system control coprocessor is implemented but writing to it does not change the behavior of the model.
- The model cannot be configured with a 128-entry TLB.
- When modeling the SCU, coherency operations are represented by a memory write followed by a read to refill from memory, rather than using cache-to-cache transfers.
- TCMs are modeled internally and the model does not support external TCMs or the ports associated with them.
- This component implements L1 cache as architecturally defined, but does not implement L2 cache. If you require an L2 cache you can add a PL310 Level 2 Cache Controller component.
- ECC and parity schemes are hardware-specific so are not supported.

Table 3-197 Ports

Name	Protocol	Type	Description
`acp_s`	`PVBus`	Slave	AXI ACP slave port.
`cfgend[1]`	*Signal* on page 2-85	Slave	This signal if for EE bit initialisation.
`cfgnmfi[1]`	*Signal* on page 2-85	Slave	This signal disables FIQ mask in CPSR.
`clk_in`	`ClockSignal`	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the core executes instructions.
`clusterid`	*Value* on page 2-85	Slave	The port reads the value in CPU ID register field, bits[11:8] of the MPIDR.
`event`	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE.
`fiq[1]`	*Signal* on page 2-85	Slave	Legacy FIQ request input line.
`fiqout[1]`	*Signal* on page 2-85	Master	Output of individual processor nFIQ from the interrupt controller.
`fpuflags[1]`	*ValueState* on page 2-85	Master	Floating-Point Unit output flags.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-197 Ports (continued)

Name	Protocol	Type	Description
halt[1]	*Signal* on page 2-85	Slave	Raising this signal will put the core into halt mode. Equivalent to the hardware nCPUHALT[N:0] signal.
initram[1]	*Signal* on page 2-85	Slave	This signal enables the processor to boot from the instruction TCM.
ints[480]	*Signal* on page 2-85	Slave	Interrupt distributor interrupt lines.
irq[1]	*Signal* on page 2-85	Slave	Legacy IRQ request input line.
irqout[1]	*Signal* on page 2-85	Master	Output of individual processor nIRQ from the interrupt controller.
mfilteren	*Signal* on page 2-85	Slave	This signal enables filtering of address ranges between master bus ports.
mfilterend	*Value* on page 2-85	Slave	This port sets end of region mapped to pvbus_m1.
mfilterstart	*Value* on page 2-85	Slave	This port sets start of region mapped to pvbus_m1.
periphbase	*Value* on page 2-85	Slave	This port sets the base address of private peripheral region.
periphclk_in	ClockSignal	Slave	The timer and the watchdog take need a clk that is scaled down atleast by factor of two.
periphreset	*Signal* on page 2-85	Slave	This signal resets timer and interrupt controller.
pfilterend	*Value* on page 2-85	Slave	This port sets end of region mapped to pvbus_mp.
pfilterstart	*Value* on page 2-85	Slave	This port sets start of region mapped to pvbus_mp.
pmuirq[2]	*Signal* on page 2-85	Master	Interrupt signal from performance monitoring unit.
pvbus_m0	PVBus	Master	The core will generate bus requests on this port.
pvbus_m1	PVBus	Master	The core will generate bus requests on this port.
pvbus_mp	PVBus	Master	The core will generate bus requests on this port.
pvbus_s	PVBus	Slave	tcm slave port
reset[1]	*Signal* on page 2-85	Slave	Raising this signal will put the core into reset mode.
scureset	*Signal* on page 2-85	Slave	This signal resets SCU.
smpnamp[1]	*Signal* on page 2-85	Master	This signals AMP or SMP mode for each Cortex-R7 processor.
standbywfe[1]	*Signal* on page 2-85	Master	This signal indicates if a core is in WFE state.
standbywfi[1]	*Signal* on page 2-85	Master	This signal indicates if a core is in WFI state.
teinit[1]	*Signal* on page 2-85	Slave	This signal provides default exception handling state.
ticks[1]	*InstructionCount* on page 2-84	Master	This port should be connected to one of the two ticks ports on a 'visualisation' component, in order to display a running instruction count.
vinithi[1]	*Signal* on page 2-85	Slave	This signal controls of the location of the exception vectors at reset.
wdreset[1]	*Signal* on page 2-85	Slave	This signal resets individual watchdog.
wdresetreq[1]	*Signal* on page 2-85	Master	CPU watchdog reset requests.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-198 Parameters for Cluster_ARM_Cortex-R7

Name	Type	Default value	Description
CLUSTER_ID	int	0x0	Processor cluster ID value
LOCK_STEP	int	0x0	Affects dual-processor configurations only, and ignored by single-processor configurations. 0 - Disable. Set for two independent processors. 1 - Lock Step. Appears to the system as two processors but is internally modeled as a single processor. 3 - Split Lock. Appears to the system as two processors but can be statically configured from reset either as two independent processors or two locked processors. For the model, these are equivalent to Disable and Lock Step, respectively, except for the value of build options registers. The model does not support dynamically splitting and locking the processor.
MFILTEREN	bool	0x0	Enables filtering of address ranges
MFILTEREND	int	0x0	Specifies the end address for address filtering
MFILTERSTART	int	0x0	Specifies the start address for address filtering
NUM_MPU_REGION	int	0xc	Sets the number of MPU regions
PERIPHBASE	int	0xae00 0000	Base address of peripheral memory space
PFILTEREND	int	0x0	Specifies the end address for peripheral port address filtering
PFILTERSTART	int	0xfff0 0000	Specifies the start address for peripheral port address filtering
cpi_div	int	0x1	Divider for calculating CPI (Cycles Per Instruction)
cpi_mul	int	0x1	Multiplier for calculating CPI (Cycles Per Instruction)
dcache-state_modelled	bool	0x0	Set whether D-cache has stateful implementation
dic-spi_count	int	0x40	Number of shared peripheral interrupts implemented
ecc_on	bool	0x0	Enable Error Correcting Code
icache-state_modelled	bool	0x0	Set whether I-cache has stateful implementation
scheduler_mode	int	0x0	Control the interleaving of instructions in this processor. 0, default long quantum. 1, low latency mode, short quantum and signal checking. 2, lock-breaking mode, long quantum with additional context switches near load-exclusive instructions.

Table 3-199 Parameters for ARM_Cortex-R7

Name	Type	Default value	Description
cpu0.CFGEND	bool	0x0	Initialize to BE8 endianness
cpu0.CFGNMFI	bool	0x0	Enable nonmaskable FIQ interrupts on startup
cpu0.DP_FLOAT	bool	0x1	Sets whether double-precision instructions are available
cpu0.INITRAM	bool	0x0	Enable the processor to boot from the instruction TCM
cpu0.POWERCTLI	int	0x0	Default power control state for processor
cpu0.SMPnAMP	bool	0x0	Set whether the processor is part of a coherent domain

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-199 Parameters for ARM_Cortex-R7 (continued)

Name	Type	Default value	Description
cpu0.TEINIT	bool	0x0	T32 exception enable. The default has exceptions including reset handled in A32 state
cpu0.VINITHI	bool	0x0	Initialize with high vectors enabled
cpu0.dcache-size	int	0x8000	Set D-cache size in bytes
cpu0.dtcm_size	int	0x8	DTCM size in KB
cpu0.icache-size	int	0x8000	Set I-cache size in bytes
cpu0.itcm_size	int	0x8	ITCM size in KB
cpu0.min_sync_level	int	0x0	Force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)
cpu0.semihosting-ARM_HLT	int	0xf000	ARM HLT number for semihosting
cpu0.semihosting-ARM_SVC	int	0x123456	ARM SVC number for semihosting
cpu0.semihosting-Thumb_HLT	int	0x3c	Thumb HLT number for semihosting
cpu0.semihosting-Thumb_SVC	int	0xab	Thumb SVC number for semihosting
cpu0.semihosting-cmd_line	string	""	Command line available to semihosting SVC calls
cpu0.semihosting-cwd	string	""	Base directory for semihosting file access.
cpu0.semihosting-enable	bool	0x1	Enable semihosting SVC traps. Applications that do not use semihosting must set this parameter to false
cpu0.semihosting-heap_base	int	0x0	Virtual address of heap base
cpu0.semihosting-heap_limit	int	0xf000000	Virtual address of top of heap
cpu0.semihosting-hlt-enable	bool	0x0	Enable semihosting HLT traps. Applications that use HLT semihosting must set this parameter to true and the semihosting-enable parameter to true
cpu0.semihosting-prefix	bool	0x0	Prefix semihosting output with target instance name
cpu0.semihosting-stack_base	int	0x10000000	Virtual address of base of descending stack
cpu0.semihosting-stack_limit	int	0xf000000	Virtual address of stack limit
cpu0.tcm-present	bool	0x1	Disables the DTCM and ITCM
cpu0.vfp-enable_at_reset	bool	0x0	Enable coprocessor access and VFP at reset
cpu0.vfp-present	bool	0x1	Set whether model has VFP support

3.4.43 ARMCortexR8x1CT

ARMCortexR8x1CT CPU component. This model is written in C++ and models version r0p0 of the RTL.

ARMCortexR8x1CT contains the following CADI targets:

- ARM_Cortex-R8
- Cluster_ARM_Cortex-R8
- PVCache

ARMCortexR8x1CT contains the following MTI components:

- *4.41 ARM_Cortex-R8 - trace* on page 4-2161
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.108 PVCache - trace* on page 4-2761

About ARMCortexR8x1CT

- The following components also exist:
  — ARMCortexR8x2CT.
  — ARMCortexR8x3CT.
  — ARMCortexR8x4CT.

  The per-core parameters are preceded by `cpu`*n*`.`, where *n* identifies the core (0-3).
- `pvbus_s` is the slave port to access the TCM RAM of CPU *n*. Bits [3:0] of the user flags in the transaction are used to select the TCM:
  — 0 selects the ITCM of CPU 0.
  — 1 selects the DTCM of CPU 0.
  — 2 selects the ITCM of CPU 1.
  — 3 selects the DTCM of CPU 1.
  — 4 selects the ITCM of CPU 2.
  — 5 selects the DTCM of CPU 2.
  — 6 selects the ITCM of CPU 3.
  — 7 selects the DTCM of CPU 3.
  — Any other value is reserved.
- The `semihosting-cwd` parameter sets the CWD that is used for semihosting. The host operating system limits the maximum path length. The `semihosting-cwd` parameter does not provide any security. Software running on the model can access files outside this directory using relative paths containing `..` or using absolute paths.

Differences between the model and the RTL

This component has the following differences from the corresponding revision of the RTL implementation:

- This component does not implement address filtering within the SCU. The enable bit for this feature is ignored.
- The GIC does not respect the CFGSDISABLE signal. This leads to some registers being accessible when they must not be.
- The SCU enable bit is ignored. The SCU is always enabled.
- The SCU ignores the invalidate all register.
- The Broadcast TLB or cache operations in this model do not cause other cores in the cluster that are asleep because of WFI to wake up.
- The RR bit in the SCTLR is ignored.
- The Power Control Register in the system control coprocessor is implemented but writing to it does not change the behavior of the model.
- The model cannot be configured with a 128-entry TLB.
- When modeling the SCU, coherency operations are represented by a memory write followed by a read to refill from memory, rather than using cache-to-cache transfers.
- TCMs are modeled internally and the model does not support external TCMs or the ports associated with them.
- This component implements L1 cache as architecturally defined, but does not implement L2 cache. If you require an L2 cache you can add a PL310 Level 2 Cache Controller component.
- The `vfp-enable_at_reset` option is model-specific and has no hardware equivalent. Arm recommends that it is only used in test systems and tied off to `false` in production systems.
- ECC and parity schemes are hardware-specific so are not supported.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-200 Ports

Name	Protocol	Type	Description
acp_s	PVBus	Slave	AXI ACP slave port.
cfgend[1]	*Signal* on page 2-85	Slave	This signal if for EE bit initialisation.
cfgnmfi[1]	*Signal* on page 2-85	Slave	This signal disables FIQ mask in CPSR.
clk_in	ClockSignal	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the core executes instructions.
clusterid	*Value* on page 2-85	Slave	The port reads the value in CPU ID register field, bits[11:8] of the MPIDR.
event	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE.
fiq[1]	*Signal* on page 2-85	Slave	Legacy FIQ request input line.
fiqout[1]	*Signal* on page 2-85	Master	Output of individual processor nFIQ from the interrupt controller.
fpfilterend0	*Value* on page 2-85	Slave	This port sets end of region mapped to pvbus_mfp0.
fpfilterend1	*Value* on page 2-85	Slave	This port sets end of region mapped to pvbus_mfp1.
fpfilterend2	*Value* on page 2-85	Slave	This port sets end of region mapped to pvbus_mfp2.
fpfilterend3	*Value* on page 2-85	Slave	This port sets end of region mapped to pvbus_mfp3.
fpfilterstart0	*Value* on page 2-85	Slave	This port sets start of region mapped to pvbus_mfp0.
fpfilterstart1	*Value* on page 2-85	Slave	This port sets start of region mapped to pvbus_mfp1.
fpfilterstart2	*Value* on page 2-85	Slave	This port sets start of region mapped to pvbus_mfp2.
fpfilterstart3	*Value* on page 2-85	Slave	This port sets start of region mapped to pvbus_mfp3.
fpuflags[1]	*ValueState* on page 2-85	Master	Floating-Point Unit output flags.
halt[1]	*Signal* on page 2-85	Slave	Raising this signal will put the core into halt mode. Equivalent to the hardware nCPUHALT[N:0] signal.
initram[1]	*Signal* on page 2-85	Slave	This signal enables the processor to boot from the instruction TCM.
ints[480]	*Signal* on page 2-85	Slave	Interrupt distributor interrupt lines.
irq[1]	*Signal* on page 2-85	Slave	Legacy IRQ request input line.
irqout[1]	*Signal* on page 2-85	Master	Output of individual processor nIRQ from the interrupt controller.
mfilteren	*Signal* on page 2-85	Slave	This signal enables filtering of address ranges between master bus ports.
mfilterend	*Value* on page 2-85	Slave	This port sets end of region mapped to pvbus_m1.
mfilterstart	*Value* on page 2-85	Slave	This port sets start of region mapped to pvbus_m1.
periphbase	*Value* on page 2-85	Slave	This port sets the base address of private peripheral region.
periphclk_in	ClockSignal	Slave	The timer and the watchdog take need a clk that is scaled down atleast by factor of two.
periphreset	*Signal* on page 2-85	Slave	This signal resets timer and interrupt controller.
pfilterend	*Value* on page 2-85	Slave	This port sets end of region mapped to pvbus_mp.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-200 Ports (continued)

Name	Protocol	Type	Description
pfilterstart	*Value* on page 2-85	Slave	This port sets start of region mapped to pvbus_mp.
pmuirq[2]	*Signal* on page 2-85	Master	Interrupt signal from performance monitoring unit.
pvbus_m0	PVBus	Master	AXI master port 0.
pvbus_m1	PVBus	Master	AXI master port 1.
pvbus_mfp0	PVBus	Master	Fast peripheral port for core 0.
pvbus_mfp1	PVBus	Master	Fast peripheral port for core 1.
pvbus_mfp2	PVBus	Master	Fast peripheral port for core 2.
pvbus_mfp3	PVBus	Master	Fast peripheral port for core 3.
pvbus_mp	PVBus	Master	Shared peripheral port.
pvbus_s	PVBus	Slave	tcm slave port.
reset[1]	*Signal* on page 2-85	Slave	Raising this signal will put the core into reset mode.
scureset	*Signal* on page 2-85	Slave	This signal resets SCU.
smpnamp[1]	*Signal* on page 2-85	Master	This signals AMP or SMP mode for each Cortex-R8 processor.
standbywfe[1]	*Signal* on page 2-85	Master	This signal indicates if a core is in WFE state.
standbywfi[1]	*Signal* on page 2-85	Master	This signal indicates if a core is in WFI state.
teinit[1]	*Signal* on page 2-85	Slave	This signal provides default exception handling state.
ticks[1]	*InstructionCount* on page 2-84	Master	This port should be connected to one of the two ticks ports on a 'visualisation' component, in order to display a running instruction count.
vinithi[1]	*Signal* on page 2-85	Slave	This signal controls of the location of the exception vectors at reset.
wdreset[1]	*Signal* on page 2-85	Slave	This signal resets individual watchdog.
wdresetreq[1]	*Signal* on page 2-85	Master	CPU watchdog reset requests.

Table 3-201 Parameters for ARM_Cortex-R8

Name	Type	Default value	Description
cpu0.CFGEND	bool	0x0	Initialize to BE8 endianness
cpu0.CFGNMFI	bool	0x0	Enable nonmaskable FIQ interrupts on startup
cpu0.DP_FLOAT	bool	0x1	Sets whether double-precision instructions are available
cpu0.INITRAM	bool	0x0	Enable the processor to boot from the instruction TCM
cpu0.POWERCTLI	int	0x0	Default power control state for processor
cpu0.SMPnAMP	bool	0x0	Set whether the processor is part of a coherent domain
cpu0.TEINIT	bool	0x0	T32 exception enable. The default has exceptions including reset handled in A32 state
cpu0.VINITHI	bool	0x0	Initialize with high vectors enabled
cpu0.dcache-size	int	0x8000	Set D-cache size in bytes

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-201 Parameters for ARM_Cortex-R8 (continued)

Name	Type	Default value	Description
cpu0.dtcm_size	int	0x8	DTCM size in KB
cpu0.icache-size	int	0x8000	Set I-cache size in bytes
cpu0.itcm_size	int	0x8	ITCM size in KB
cpu0.min_sync_level	int	0x0	force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)
cpu0.semihosting-ARM_HLT	int	0xf000	ARM HLT number for semihosting
cpu0.semihosting-ARM_SVC	int	0x123456	ARM SVC number for semihosting
cpu0.semihosting-Thumb_HLT	int	0x3c	Thumb HLT number for semihosting
cpu0.semihosting-Thumb_SVC	int	0xab	Thumb SVC number for semihosting
cpu0.semihosting-cmd_line	string	""	Command line available to semihosting SVC calls
cpu0.semihosting-cwd	string	""	Base directory for semihosting file access.
cpu0.semihosting-enable	bool	0x1	Enable semihosting SVC traps. Applications that do not use semihosting must set this parameter to false
cpu0.semihosting-heap_base	int	0x0	Virtual address of heap base
cpu0.semihosting-heap_limit	int	0xf000000	Virtual address of top of heap
cpu0.semihosting-hlt-enable	bool	0x0	Enable semihosting HLT traps. Applications that use HLT semihosting must set this parameter to true and the semihosting-enable parameter to true
cpu0.semihosting-prefix	bool	0x0	Prefix semihosting output with target instance name
cpu0.semihosting-stack_base	int	0x10000000	Virtual address of base of descending stack
cpu0.semihosting-stack_limit	int	0xf000000	Virtual address of stack limit
cpu0.tcm-present	bool	0x1	Disables the DTCM and ITCM
cpu0.vfp-enable_at_reset	bool	0x0	Enable coprocessor access and VFP at reset
cpu0.vfp-present	bool	0x1	Set whether model has VFP support

Table 3-202 Parameters for Cluster_ARM_Cortex-R8

Name	Type	Default value	Description
CLUSTER_ID	int	0x0	Processor cluster ID value
FPFILTEREND0	int	0x0	Specifies the end address for the fast peripheral port address filtering
FPFILTEREND1	int	0x0	Specifies the end address for the fast peripheral port address filtering
FPFILTEREND2	int	0x0	Specifies the end address for the fast peripheral port address filtering
FPFILTEREND3	int	0x0	Specifies the end address for the fast peripheral port address filtering

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-202 Parameters for Cluster_ARM_Cortex-R8 (continued)

Name	Type	Default value	Description
FPFILTERSTART0	int	0xfff00000	Specifies the start address for the fast peripheral port address filtering
FPFILTERSTART1	int	0xfff00000	Specifies the start address for the fast peripheral port address filtering
FPFILTERSTART2	int	0xfff00000	Specifies the start address for the fast peripheral port address filtering
FPFILTERSTART3	int	0xfff00000	Specifies the start address for the fast peripheral port address filtering
LOCK_STEP	int	0x0	Affects dual-processor configurations only, and ignored by single-processor configurations. 0 - Disable. Set for two independent processors. 1 - Lock Step. Appears to the system as two processors but is internally modeled as a single processor. 3 - Split Lock. Appears to the system as two processors but can be statically configured from reset either as two independent processors or two locked processors. For the model, these are equivalent to Disable and Lock Step, respectively, except for the value of build options registers. The model does not support dynamically splitting and locking the processor.
MFILTEREN	bool	0x0	Enables filtering of address ranges
MFILTEREND	int	0x0	Specifies the end address for address filtering
MFILTERSTART	int	0x0	Specifies the start address for address filtering
NUM_MPU_REGION	int	0xc	Sets the number of MPU regions
PERIPHBASE	int	0xae000000	Base address of peripheral memory space
PFILTEREND	int	0x0	Specifies the end address for peripheral port address filtering
PFILTERSTART	int	0xfff00000	Specifies the start address for peripheral port address filtering
cpi_div	int	0x1	Divider for calculating CPI (Cycles Per Instruction)
cpi_mul	int	0x1	Multiplier for calculating CPI (Cycles Per Instruction)
dcache-state_modelled	bool	0x0	Set whether D-cache has stateful implementation
dic-spi_count	int	0x40	Number of shared peripheral interrupts implemented
ecc_on	bool	0x0	Enable Error Correcting Code
icache-state_modelled	bool	0x0	Set whether I-cache has stateful implementation
scheduler_mode	int	0x0	Control the interleaving of instructions in this processor. 0, default long quantum. 1, low latency mode, short quantum and signal checking. 2, lock-breaking mode, long quantum with additional context switches near load-exclusive instructions.

3.4.44 ARMCortexR52x1CT

ARMCortexR52x1CT CPU component. This model is written in C++.

ARMCortexR52x1CT contains the following CADI targets:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

- ARM_CortexR52
- PVCache
- TlbCadi
- gic_iri

ARMCortexR52x1CT contains the following MTI components:

- *4.43 ARM_CortexR52 - trace* on page 4-2235
- *4.48 ARMv8Cluster - trace* on page 4-2401
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402
- *4.72 GICv3CPUInterface - trace* on page 4-2474
- *4.73 GICv3CPUInterfaceDecoder - trace* on page 4-2489
- *4.76 GICv3Distributor - trace* on page 4-2496
- *4.77 GICv3IRI - trace* on page 4-2504
- *4.80 GICv3Redistributor - trace* on page 4-2527
- *4.81 GICv3RedistributorInternal - trace* on page 4-2541
- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.108 PVCache - trace* on page 4-2761
- *4.114 TLB - trace* on page 4-2834

About ARMCortexR52x1CT

- The following components also exist:
  - ARMCortexR52x2CT.
  - ARMCortexR52x3CT.
  - ARMCortexR52x4CT.

  The per-core parameters are preceded by `cpun.`, where *n* identifies the core (0-3).
- When you use the `semihosting-cmd_line` parameter to set the command line that is available to semihosting SVC calls, the value of `argv[0]` is set to the first command-line argument, not to the name of an image.
- The Cortex-R52 processor does not implement TrustZone technology, therefore the model does not support S_* or NS_* registers or exceptions.

Differences between the model and the RTL

- The model does not implement redundant cores for Dual-Core Lock-Step operations.
- The model does not implement the Low Power Interface to wake up the target core on receiving a `wake_request` signal from the GIC distributor.
- TCMs are modeled internally and the model does not support external TCMs or the ports associated with them.
- The model does not support running Software Test Libraries (STLs).
- The `vfp-enable_at_reset` parameter is a model-specific behavior with no hardware equivalent.
- ECC and parity schemes are hardware-specific so are not supported.

Table 3-203 Ports

Name	Protocol	Type	Description
`cfgdbgromaddr`	*`Value_64` on page 2-85*	Slave	Debug ROM base address.
`cfgdbgromaddrv`	*`Signal` on page 2-85*	Slave	Debug ROM base address valid.
`cfgendianess[1]`	*`Signal` on page 2-85*	Slave	This signal if for EE bit initialisation.
`cfgperiphbase`	*`Value_64` on page 2-85*	Slave	This port sets the base address of private peripheral region.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-203 Ports (continued)

Name	Protocol	Type	Description
cfgthumbexceptions[1]	*Signal* on page 2-85	Slave	This signal provides default exception handling state.
cfgvectable[1]	*Value_64* on page 2-85	Slave	Reset vector base address.
clk_in	ClockSignal	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the core executes instructions.
clrexmonack	*Signal* on page 2-85	Master	Acknowledge handshake signal for the clrexmonreq signal
clrexmonreq	*Signal* on page 2-85	Slave	Signals the clearing of an external global exclusive monitor
cntvalueb	*CounterInterface* on page 2-84	Slave	Interface to SoC level counter module.
commrx[1]	*Signal* on page 2-85	Master	Receive portion of Data Transfer Register full.
commtx[1]	*Signal* on page 2-85	Master	Transmit portion of Data Transfer Register empty.
cpuhalt[1]	*Signal* on page 2-85	Slave	Raising this signal will put the core into halt mode.
cpuporeset[1]	*Signal* on page 2-85	Slave	Power on reset. Initializes all the processor logic, including debug logic.
cti[1]	*v8EmbeddedCrossTrigger_controlprotocol* on page 2-84	Master	Cross trigger matrix port.
dbgack[1]	*Signal* on page 2-85	Master	External debug interface.
dbgen[1]	*Signal* on page 2-85	Slave	External debug interface.
dbgnopwrdwn[1]	*Signal* on page 2-85	Master	There is no support for PChannel in CortexR52. These signals relate to core power down. Equivalent to COREPACTIVEx
dbgpwrupreq[1]	*Signal* on page 2-85	Master	Debug power up request.
dev_debug_s	PVBus	Slave	External debug interface.
edbgrq[1]	*Signal* on page 2-85	Slave	External debug interface.
event	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE.
ext_slave_s	PVBus	Slave	External Slave port. Equivalent to AXIS port
extppi_in_0[9]	*Signal* on page 2-85	Slave	Core 0 external ppi signals.
flash_m[1]	PVBus	Master	Flash Port.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-203 Ports (continued)

Name	Protocol	Type	Description
gdu_external_m	*GICv3Comms* on page 2-84	Master	GDU external messaging port.
hiden[1]	*Signal* on page 2-85	Slave	External debug interface.
hniden[1]	*Signal* on page 2-85	Slave	External debug interface.
llpp_m[1]	PVBus	Master	LLPP (Low-Latency Peripheral Port).
memorymapped_debug_s	PVBus	Slave	External debug interface.
niden[1]	*Signal* on page 2-85	Slave	External debug interface.
presetdbg	*Signal* on page 2-85	Slave	Initialize the shared debug APB, Cross Trigger Interface (CTI), and Cross Trigger Matrix (CTM) logic.
pvbus_core_m[1]	PVBus	Master	The core will generate bus requests on this port. Equivalent to AXIM port
reset[1]	*Signal* on page 2-85	Slave	Raising this signal will put the core into reset mode.
sei[1]	*Signal* on page 2-85	Slave	Per core virtual System Error physical pins.
spi_in[960]	*Signal* on page 2-85	Slave	Shared peripheral interrupts.
standbywfe[1]	*Signal* on page 2-85	Master	This signal indicates if a core is in WFE state.
standbywfi[1]	*Signal* on page 2-85	Master	This signal indicates if a core is in WFI state.
ticks[1]	*InstructionCount* on page 2-84	Master	This port should be connected to one of the two ticks ports on a 'visualisation' component, in order to display a running instruction count.
topreset	*Signal* on page 2-85	Slave	This signal resets timer and interrupt controller.
vsei[1]	*Signal* on page 2-85	Slave	Per core virtual System Error physical pins.
warmrstreq[1]	*Signal* on page 2-85	Master	Warm reset request from core.

Table 3-204 Parameters for ARM_CortexR52

Name	Type	Default value	Description
CLUSTER_ID	int	0x0	CLUSTER_ID[15:8] equivalent to CFGMPIDRAFF2, CLUSTER_ID[7:0] equivalent to CFGMPIDRAFF1
DBGROMADDR	int	0x0	Equivalent to CFGDBGROMADDR

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-204 Parameters for ARM_CortexR52 (continued)

Name	Type	Default value	Description
DBGROMADDRV	bool	0x0	If true, set bits[1:0] of the CP15 DBGDRAR to indicate that the address is valid
GICDISABLE	bool	0x0	Disable the new style GICv3 CPU interface in each core model. Should be left enabled unless the platform contains a GICv3 distributor.
PERIPHBASE	int	0x13080000	Equivalent to CFGPERIPHBASE
cluster_utid	int	0x0	Equivalent to CFGCLUSTERUTID
cpi_div	int	0x1	Divider for calculating CPI (Cycles Per Instruction)
cpi_mul	int	0x1	Multiplier for calculating CPI (Cycles Per Instruction)
dcache-prefetch_enabled	bool	0x0	Enable simulation of data cache prefetching. This is only used when dcache-state_modelled=true
dcache-read_access_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per access (of size dcache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
dcache-state_modelled	bool	0x0	Set whether D-cache has stateful implementation
dcache-write_access_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per access (of size dcache-write_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-write_latency is set. This is only used when dcache-state_modelled=true.
enable_simulation_performance_optimizations	bool	0x1	With this option enabled, the model will run more quickly, but be less accurate to exact CPU behavior. The model will still be functionally accurate for software, but may increase differences seen between hardware behavior and model behavior for certain workloads (it changes the micro-architectural value of stage12_tlb_size parameter to 1024).
flash_protection_enable_at_reset	bool	0x0	Equivalent to CFGFLASHPROTEN
has_export_m_port	bool	0x1	The interrupt distributor has an optional interrupt export port for routing interrupts to an external device
has_flash	bool	0x0	Equivalent to CFGFLASHIMP

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-204 Parameters for ARM_CortexR52 (continued)

Name	Type	Default value	Description
has_flash_protection	bool	0x1	Equivalent to CFGFLASHPROTIMP
icache-prefetch_enabled	bool	0x0	Enable simulation of instruction cache prefetching. This is only used when icache-state_modelled=true.
icache-read_access_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per access (of size icache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if icache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when icache-state_modelled=true.
icache-state_modelled	bool	0x0	Set whether I-cache has stateful implementation
memory.ext_slave_base	int	0x0	Equivalent to CFGAXISTCMBASEADDR
memory.flash_base	int	0x0	Equivalent to CFGFLASHBASEADDR
num_protection_regions_s1	int	0x10	Number of v8-R protection regions
num_protection_regions_s2	int	0x10	Number of v8-R hyp protection regions
num_spi	int	0x3c0	Number of interrupts (SPI) into the internal GIC controller
ram_protection_enable_at_reset	bool	0x0	Equivalent to CFGRAMPROTEN

Table 3-205 Parameters for gic_iri

Name	Type	Default value	Description
gic_iri.A3-affinity-supported	bool	0x0	Device supports affinity level 3 values that are non-zero.
gic_iri.ARE-fixed-to-one	bool	0x0	GICv2 compatibility is not supported and GICD_CTLR.ARE_* is always one
gic_iri.CPU-affinities	string	""	A comma separated list of dotted quads containing the affinities of all PEs connected to this IRI. If CPU-affinities-file is specified, this parameter is ignored.
gic_iri.CPU-affinities-file	string	""	A file containing comma separated list of dotted quads containing the affinities of all PEs connected to this IRI. If this parameter is specified, CPU-affinities parameter will be ignored even when it is given.
gic_iri.DPG-ARE-only	bool	0x0	Limit application of DPG bits to interrupt groups for which ARE=1
gic_iri.DPG-bits-implemented	bool	0x0	Enable implementation of interrupt group participation bits or DPG bits in GICR_CTLR
gic_iri.DS-fixed-to-zero	bool	0x0	Enable/disable support of single security state

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-205 Parameters for gic_iri (continued)

Name	Type	Default value	Description
gic_iri.GICD-alias	int	0x0	In GICv2 mode: the base address for a 4k page alias of the first 4k of the Distributor page, in GICv3 mode. the base address of a 64KB page containing message based SPI signalling register aliases(0:Disabled)
gic_iri.GICD-legacy-registers-as-reserved	bool	0x0	When ARE is RAO/WI, makes superfluous registers in GICD reserved (including for the purpose of STATUSR updates)
gic_iri.GICD_CTLR-DS-1-means-secure-only	bool	0x0	If GICD_CTLR.DS=1, GICD supports a single security state which is secure if this is true, otherwise is non-secure.
gic_iri.GICD_ITARGETSR-RAZWI	bool	0x0	If true, the GICD_ITARGETS registers are RAZ/WI
gic_iri.GICD_PIDR	int	0x0	The value for the GICD_PIDR registers, if non-zero. Note: fixed fields (device type etc.) will be overriden in this value.
gic_iri.GICR-clear-enable-supported	bool	0x0	When true, this sets the value of the RO bit GICR_CTLR.CES with the value of the parameter allow-LPIEN-clear, making it visible to software
gic_iri.GICR-invalidate-registers-implemented	bool	0x0	When true, the registers GICR_INVLPIR, GICR_INVALLR and GICR_SYNCR are implemented
gic_iri.GICR_PIDR	int	0x0	The value for the GICR_PIDR registers, if non-zero. Note: fixed fields (device type etc.) will be overriden in this value.
gic_iri.GICR_PROPBASER-read-only	bool	0x0	GICR_PROPBASER register is read-only.
gic_iri.GICR_PROPBASER-reset-value	int	0x0	Value of GICR_PROPBASER on reset.
gic_iri.GITS_BASER0-entry-bytes	int	0x8	Number of bytes required per entry for GITS_BASER0 register.
gic_iri.GITS_BASER0-indirect-RAZ	bool	0x0	Indirect field for GITS_BASER0 register is RAZ/WI.
gic_iri.GITS_BASER0-type	int	0x0	Type field for GITS_BASER0 register. 0 = Unimplemented; 1 = Devices; 2 = Virtual Processors; 3 = Physical Processors; 4 = Collections
gic_iri.GITS_BASER1-entry-bytes	int	0x8	Number of bytes required per entry for GITS_BASER1 register.
gic_iri.GITS_BASER1-indirect-RAZ	bool	0x0	Indirect field for GITS_BASER1 register is RAZ/WI.
gic_iri.GITS_BASER1-type	int	0x0	Type field for GITS_BASER1 register. 0 = Unimplemented; 1 = Devices; 2 = Virtual Processors; 3 = Physical Processors; 4 = Collections
gic_iri.GITS_BASER2-entry-bytes	int	0x8	Number of bytes required per entry for GITS_BASER2 register.
gic_iri.GITS_BASER2-indirect-RAZ	bool	0x0	Indirect field for GITS_BASER2 register is RAZ/WI.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-205 Parameters for gic_iri (continued)

Name	Type	Default value	Description
gic_iri.GITS_BASER2-type	int	0x0	Type field for GITS_BASER2 register. 0 = Unimplemented; 1 = Devices; 2 = Virtual Processors; 3 = Physical Processors; 4 = Collections
gic_iri.GITS_BASER3-entry-bytes	int	0x8	Number of bytes required per entry for GITS_BASER3 register.
gic_iri.GITS_BASER3-indirect-RAZ	bool	0x0	Indirect field for GITS_BASER3 register is RAZ/WI.
gic_iri.GITS_BASER3-type	int	0x0	Type field for GITS_BASER3 register. 0 = Unimplemented; 1 = Devices; 2 = Virtual Processors; 3 = Physical Processors; 4 = Collections
gic_iri.GITS_BASER4-entry-bytes	int	0x8	Number of bytes required per entry for GITS_BASER4 register.
gic_iri.GITS_BASER4-indirect-RAZ	bool	0x0	Indirect field for GITS_BASER4 register is RAZ/WI.
gic_iri.GITS_BASER4-type	int	0x0	Type field for GITS_BASER4 register. 0 = Unimplemented; 1 = Devices; 2 = Virtual Processors; 3 = Physical Processors; 4 = Collections
gic_iri.GITS_BASER5-entry-bytes	int	0x8	Number of bytes required per entry for GITS_BASER5 register.
gic_iri.GITS_BASER5-indirect-RAZ	bool	0x0	Indirect field for GITS_BASER5 register is RAZ/WI.
gic_iri.GITS_BASER5-type	int	0x0	Type field for GITS_BASER5 register. 0 = Unimplemented; 1 = Devices; 2 = Virtual Processors; 3 = Physical Processors; 4 = Collections
gic_iri.GITS_BASER6-entry-bytes	int	0x8	Number of bytes required per entry for GITS_BASER6 register.
gic_iri.GITS_BASER6-indirect-RAZ	bool	0x0	Indirect field for GITS_BASER6 register is RAZ/WI.
gic_iri.GITS_BASER6-type	int	0x0	Type field for GITS_BASER6 register. 0 = Unimplemented; 1 = Devices; 2 = Virtual Processors; 3 = Physical Processors; 4 = Collections
gic_iri.GITS_BASER7-entry-bytes	int	0x8	Number of bytes required per entry for GITS_BASER7 register.
gic_iri.GITS_BASER7-indirect-RAZ	bool	0x0	Indirect field for GITS_BASER7 register is RAZ/WI.
gic_iri.GITS_BASER7-type	int	0x0	Type field for GITS_BASER7 register. 0 = Unimplemented; 1 = Devices; 2 = Virtual Processors; 3 = Physical Processors; 4 = Collections
gic_iri.GITS_PIDR	int	0x0	The value for the GITS_PIDR registers, if non-zero. Note: fixed fields (device type etc.) will be overriden in this value.
gic_iri.ICFGR-PPI-mask	int	0xaaaaaaaa	Mask for writes to ICFGR registers that configure PPIs
gic_iri.ICFGR-PPI-reset	int	0x0	Reset value for ICFGR regesters that configure PPIs

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-205 Parameters for gic_iri (continued)

Name	Type	Default value	Description
gic_iri.ICFGR-SGI-mask	int	0x0	Mask for writes to ICFGR registers that configure SGIs
gic_iri.ICFGR-SGI-reset	int	0xaaaaaaaa	Reset value for ICFGR registers that configure SGIs
gic_iri.ICFGR-SPI-mask	int	0xaaaaaaaa	Mask for writes to ICFGR registers that configure SPIs
gic_iri.ICFGR-SPI-reset	int	0x0	Reset value for ICFGR regesters that configure SPIs
gic_iri.ICFGR-rsvd-bit	bool	0x0	If ARE=0, the value of reserved bits i.e. bit 0,2,4..30 of ICFGRn for n>0
gic_iri.IGROUP-PPI-mask	int	0xffff	Mask for writes to PPI bits in IGROUP registers
gic_iri.IGROUP-PPI-reset	int	0x0	Reset value for SGI bits in IGROUP registers
gic_iri.IGROUP-SGI-mask	int	0xffff	Mask for writes to SGI bits in IGROUP registers
gic_iri.IGROUP-SGI-reset	int	0x0	Reset value for SGI bits in IGROUP registers
gic_iri.IIDR	int	0x0	GICD_IIDR and GICR_IIDR value
gic_iri.IRI-ID-bits	int	0x10	Number of bits used to represent interrupts IDs in the Distributor and Redistributors, forced to 10 when none of LPIs, extended SPIs or extended PPIs is supported
gic_iri.IROUTER-IRM-RAZ-WI	bool	0x0	GICD_IROUTERn.InterruptRoutingMode is RAZ/WI
gic_iri.ITS-BASER-force-page-alignement	bool	0x1	Force alignement of address writen to a GITS_BASER register to the page size configured
gic_iri.ITS-ID-bits	int	0x10	Number of interrupt bits supported by ITS.
gic_iri.ITS-MOVALL-update-collections	bool	0x0	Whether MOVALL command updates the collection entires
gic_iri.ITS-TRANSLATE64R	bool	0x0	Add an implementation specific register at 0x10008 supporting 64 bit TRANSLATER (dev[63:32], interupt[31:0])
gic_iri.ITS-collection-ID-bits	int	0x0	Number of collection bits supported by ITS (optional parameter, 0 => 16bits support and GITS_TYPER.CIL=0
gic_iri.ITS-count	int	0x0	Number of Interrupt Translation Services to be instantiated (0=none)
gic_iri.ITS-cumulative-collection-tables	bool	0x1	When true, the supported amount of collections is the sum of GITS_TYPER.HCC and the number of collections supported in memory, otherwise, simply the number supported in memory only. Irrelevant when HCC=0
gic_iri.ITS-device-bits	int	0x10	Number of bits supported for ITS device IDs.
gic_iri.ITS-entry-size	int	0x8	Number of bytes required to store each entry in the ITT tables.
gic_iri.ITS-hardware-collection-count	int	0x0	Number of hardware collections held exclusively in the ITS
gic_iri.ITS-legacy-iidr-typer-offset	bool	0x0	Put the GITS_IIDR and GITS_TYPER registers at their older offset of 0x8 and 0x4 respectively
gic_iri.ITS-shared-vPE-table	int	0x0	Number of affinity levels to which the vPE configuration table is shared. This parameter is valid when has-gicv4.1 is true.
gic_iri.ITS-threaded-command-queue	bool	0x1	Enable execution of ITS commands in a separate thread which is sometimes required for cosimulation

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-205 Parameters for gic_iri (continued)

Name	Type	Default value	Description
gic_iri.ITS-use-physical-target-addresses	bool	0x1	Use physical hardware adresses for targets in ITS commands -- must be true for distributed implementations
gic_iri.ITS-vmovp-bit	bool	0x0	Device supports software issuing a VMOVP to only one of the ITSs that has a mapping for a vPE. The device itself ensures synchronization of the VMOVP command across all ITSs that have mapping for that vPE.
gic_iri.ITS0-base	int	0x0	Register base address for ITS0 (automatic if 0).
gic_iri.ITS1-base	int	0x0	Register base address for ITS1 (automatic if 0).
gic_iri.ITS2-base	int	0x0	Register base address for ITS2 (automatic if 0).
gic_iri.ITS3-base	int	0x0	Register base address for ITS3 (automatic if 0).
gic_iri.LPI-cache-check-data	bool	0x0	Enable Cached LPI data against memory checking when available for cache type
gic_iri.LPI-cache-type	int	0x1	Cache type for LPIs, 0:No caching, 1:Full caching
gic_iri.MSI_IIDR	int	0x0	Value returned in MSI_IIDR registers.
gic_iri.MSI_NS-frame0-base	int	0x0	If non-zero, sets the base address used for non-secure MSI frame 0 registers.
gic_iri.MSI_NS-frame0-max-SPI	int	0x0	Maximum SPI ID supported by non-secure MSI frame 0. Set to 0 to disable frame.
gic_iri.MSI_NS-frame0-min-SPI	int	0x0	Minimum SPI ID supported by non-secure MSI frame 0. Set to 0 to disable frame.
gic_iri.MSI_NS-frame1-base	int	0x0	If non-zero, sets the base address used for non-secure MSI frame 1 registers.
gic_iri.MSI_NS-frame1-max-SPI	int	0x0	Maximum SPI ID supported by non-secure MSI frame 1. Set to 0 to disable frame.
gic_iri.MSI_NS-frame1-min-SPI	int	0x0	Minimum SPI ID supported by non-secure MSI frame 1. Set to 0 to disable frame.
gic_iri.MSI_NS-frame2-base	int	0x0	If non-zero, sets the base address used for non-secure MSI frame 2 registers.
gic_iri.MSI_NS-frame2-max-SPI	int	0x0	Maximum SPI ID supported by non-secure MSI frame 2. Set to 0 to disable frame.
gic_iri.MSI_NS-frame2-min-SPI	int	0x0	Minimum SPI ID supported by non-secure MSI frame 2. Set to 0 to disable frame.
gic_iri.MSI_NS-frame3-base	int	0x0	If non-zero, sets the base address used for non-secure MSI frame 3 registers.
gic_iri.MSI_NS-frame3-max-SPI	int	0x0	Maximum SPI ID supported by non-secure MSI frame 3. Set to 0 to disable frame.
gic_iri.MSI_NS-frame3-min-SPI	int	0x0	Minimum SPI ID supported by non-secure MSI frame 3. Set to 0 to disable frame.
gic_iri.MSI_NS-frame4-base	int	0x0	If non-zero, sets the base address used for non-secure MSI frame 4 registers.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-205 Parameters for gic_iri (continued)

Name	Type	Default value	Description
gic_iri.MSI_NS-frame4-max-SPI	int	0x0	Maximum SPI ID supported by non-secure MSI frame 4. Set to 0 to disable frame.
gic_iri.MSI_NS-frame4-min-SPI	int	0x0	Minimum SPI ID supported by non-secure MSI frame 4. Set to 0 to disable frame.
gic_iri.MSI_NS-frame5-base	int	0x0	If non-zero, sets the base address used for non-secure MSI frame 5 registers.
gic_iri.MSI_NS-frame5-max-SPI	int	0x0	Maximum SPI ID supported by non-secure MSI frame 5. Set to 0 to disable frame.
gic_iri.MSI_NS-frame5-min-SPI	int	0x0	Minimum SPI ID supported by non-secure MSI frame 5. Set to 0 to disable frame.
gic_iri.MSI_NS-frame6-base	int	0x0	If non-zero, sets the base address used for non-secure MSI frame 6 registers.
gic_iri.MSI_NS-frame6-max-SPI	int	0x0	Maximum SPI ID supported by non-secure MSI frame 6. Set to 0 to disable frame.
gic_iri.MSI_NS-frame6-min-SPI	int	0x0	Minimum SPI ID supported by non-secure MSI frame 6. Set to 0 to disable frame.
gic_iri.MSI_NS-frame7-base	int	0x0	If non-zero, sets the base address used for non-secure MSI frame 7 registers.
gic_iri.MSI_NS-frame7-max-SPI	int	0x0	Maximum SPI ID supported by non-secure MSI frame 7. Set to 0 to disable frame.
gic_iri.MSI_NS-frame7-min-SPI	int	0x0	Minimum SPI ID supported by non-secure MSI frame 7. Set to 0 to disable frame.
gic_iri.MSI_PIDR	int	0x0	The value for the MSI_PIDR registers, if non-zero and distributor supports GICv2m. Note: fixed fields (device type etc.) will be overriden in this value.
gic_iri.MSI_S-frame0-base	int	0x0	If non-zero, sets the base address used for secure MSI frame 0 registers.
gic_iri.MSI_S-frame0-max-SPI	int	0x0	Maximum SPI ID supported by secure MSI frame 0. Set to 0 to disable frame.
gic_iri.MSI_S-frame0-min-SPI	int	0x0	Minimum SPI ID supported by secure MSI frame 0. Set to 0 to disable frame.
gic_iri.MSI_S-frame1-base	int	0x0	If non-zero, sets the base address used for secure MSI frame 1 registers.
gic_iri.MSI_S-frame1-max-SPI	int	0x0	Maximum SPI ID supported by secure MSI frame 1. Set to 0 to disable frame.
gic_iri.MSI_S-frame1-min-SPI	int	0x0	Minimum SPI ID supported by secure MSI frame 1. Set to 0 to disable frame.
gic_iri.MSI_S-frame2-base	int	0x0	If non-zero, sets the base address used for secure MSI frame 2 registers.
gic_iri.MSI_S-frame2-max-SPI	int	0x0	Maximum SPI ID supported by secure MSI frame 2. Set to 0 to disable frame.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-205 Parameters for gic_iri (continued)

Name	Type	Default value	Description
gic_iri.MSI_S-frame2-min-SPI	int	0x0	Minimum SPI ID supported by secure MSI frame 2. Set to 0 to disable frame.
gic_iri.MSI_S-frame3-base	int	0x0	If non-zero, sets the base address used for secure MSI frame 3 registers.
gic_iri.MSI_S-frame3-max-SPI	int	0x0	Maximum SPI ID supported by secure MSI frame 3. Set to 0 to disable frame.
gic_iri.MSI_S-frame3-min-SPI	int	0x0	Minimum SPI ID supported by secure MSI frame 3. Set to 0 to disable frame.
gic_iri.MSI_S-frame4-base	int	0x0	If non-zero, sets the base address used for secure MSI frame 4 registers.
gic_iri.MSI_S-frame4-max-SPI	int	0x0	Maximum SPI ID supported by secure MSI frame 4. Set to 0 to disable frame.
gic_iri.MSI_S-frame4-min-SPI	int	0x0	Minimum SPI ID supported by secure MSI frame 4. Set to 0 to disable frame.
gic_iri.MSI_S-frame5-base	int	0x0	If non-zero, sets the base address used for secure MSI frame 5 registers.
gic_iri.MSI_S-frame5-max-SPI	int	0x0	Maximum SPI ID supported by secure MSI frame 5. Set to 0 to disable frame.
gic_iri.MSI_S-frame5-min-SPI	int	0x0	Minimum SPI ID supported by secure MSI frame 5. Set to 0 to disable frame.
gic_iri.MSI_S-frame6-base	int	0x0	If non-zero, sets the base address used for secure MSI frame 6 registers.
gic_iri.MSI_S-frame6-max-SPI	int	0x0	Maximum SPI ID supported by secure MSI frame 6. Set to 0 to disable frame.
gic_iri.MSI_S-frame6-min-SPI	int	0x0	Minimum SPI ID supported by secure MSI frame 6. Set to 0 to disable frame.
gic_iri.MSI_S-frame7-base	int	0x0	If non-zero, sets the base address used for secure MSI frame 7 registers.
gic_iri.MSI_S-frame7-max-SPI	int	0x0	Maximum SPI ID supported by secure MSI frame 7. Set to 0 to disable frame.
gic_iri.MSI_S-frame7-min-SPI	int	0x0	Minimum SPI ID supported by secure MSI frame 7. Set to 0 to disable frame.
gic_iri.PA_SIZE	int	0x30	Number of valid bits in physical address
gic_iri.PPI-implemented-mask	int	0xffff	Mask of PPIs that are implemented. One bit per PPI bit 0 == PPI 16 (first PPI). This will affect other masks.
gic_iri.SPI-count	int	0xe0	Number of SPIs that are implemented.
gic_iri.SPI-message-based-support	bool	0x1	Distributor supports meassage based signaling of SPI
gic_iri.SPI-unimplemented	string	""	A comma spearated list of unimplemented SPIs ranges for sparse SPI defintion(for ex: '35, 39-42, 73)'
gic_iri.STATUSR-implemented	bool	0x1	Determines whether the GICR_STATUSR register is implemented.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-205 Parameters for gic_iri (continued)

Name	Type	Default value	Description
gic_iri.allow-LPIEN-clear	bool	0x0	Allow RW behaviour on GICR_CTLR.LPIEN isntead of set once
gic_iri.chip-id	int	0x0	Chip ID when multichip operation is enabled.
gic_iri.common-lpi-configuration	int	0x0	Describes which re-distributors share (and must be configured with the same) LPI configuration table as described in GICR_TYPER(0:All, 1:A.x.x.x, 2:A.B.x.x, 3:A.B.C.x
gic_iri.common-vPE-table-affinity	string	""	Affinity value list in the form of 'a.b.c.d, e.f.g.h, etc' given to the ITS(s) (where a.b.c.d corresponds to ITS0, e.f.g.h corresponds to ITS1 and so on). Under an affinity value the vPE configuration table is shared among redistributors where the level to be shared is defined by ITS-shared-vPE-table. This parameter is valid when has-gicv4.1 is true.
gic_iri.consolidators	string	""	Specify consolidators' base addresses, interrupt line counts and base interrupt IDs, in the form 'baseAddr0:itlineCount0:baseINTID0, baseAddr1:itlineCount1:baseINTID1, [etc]' (eg '0x3f100000:64:4096, 0x3f200000:64:4224'). The consolidators' count is inferred from the list (maximum of 4). If not specified, the component contains no consolidators.
gic_iri.delay-ITS-accesses	bool	0x1	Delay accesses from the ITS until GICR_SYNCR is read.
gic_iri.delay-redistributor-accesses	bool	0x1	Delay memory accesses from the redistributor until GICR_SYNCR is read.
gic_iri.direct-lpi-support	bool	0x0	Enable support for LPI operations through GICR registers
gic_iri.enable-multichip-operation	bool	0x0	Enables multi-chip operation between Distributors in distributed GIC IRI
gic_iri.enable_protocol_checking	bool	0x0	Enable/disable protocol checking at cpu interface
gic_iri.enabled	bool	0x1	Enable GICv3 functionality; when false the component is inactive.
gic_iri.extended-ppi-count	int	0x0	Number of extended PPI supported
gic_iri.extended-spi-count	int	0x0	Number of extended SPI supported
gic_iri.fixed-routed-spis	string	""	Value of IROUTER[n] register in the form 'n=a.b.c.d, n=*'. The RM bit of IROUTER is 0 when n=a.b.c.d is used else 1 when n=* is used. n can be >= 32 and <= 1019
gic_iri.gicv2-only	bool	0x0	If true, when using the GICv3 model, pretend to be a GICv2 system
gic_iri.has-two-security-states	bool	0x1	If true, has two security states
gic_iri.has_VPENDBASER-dirty-flag-on-load	bool	0x0	GICR_VPENDBASER.Dirty reflects transient loading state when valid=1
gic_iri.has_mpam	bool	0x0	Enable MPAM support on ITS and RDs
gic_iri.ignore-generate-sgi-when-no-are	bool	0x0	Ignore GenerateSGI packets coming form the CPU interface if both ARE_S and ARE_NS are 0
gic_iri.irouter-default-mask	string	""	Default Mask value for IROUTER[32..1019] register in the form 'a.b.c.d'

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-205 Parameters for gic_iri (continued)

Name	Type	Default value	Description
gic_iri.irouter-default-reset	string	""	Default Reset Value of IROUTER[32..1019] register in the form 'a.b.c.d' or *
gic_iri.irouter-mask-values	string	""	Mask Value of IROUTER[n] register in the form 'n=a.b.c.d'.n can be >= 32 and <= 1019
gic_iri.irouter-reset-values	string	""	Reset Value of IROUTER[n] register in the form 'n=a.b.c.d or n=*'.n can be >= 32 and <= 1019
gic_iri.legacy-sgi-enable-rao	bool	0x0	Enables for SGI associated with an ARE=0 regime are RAO/WI
gic_iri.local-SEIs	bool	0x0	Generate SEI to signal internal issues
gic_iri.local-VSEIs	bool	0x0	Generate VSEI to signal internal issues
gic_iri.lockable-SPI-count	int	0x0	Number of SPIs that are locked down when CFGSDISABLE signal is asserted. Only applies for GICv2.
gic_iri.max-pe-on-chip	int	0x4	Maximum number of cores on any single chip. This will be used to identify the target chip and core when multichip operation is enabled.
gic_iri.monolithic	bool	0x0	Indicate that the implementation is not distributed
gic_iri.mpam_max_partid	int	0xffff	Maximum valid PARTID
gic_iri.mpam_max_pmg	int	0xff	Maximum valid PMG
gic_iri.multichip-threaded-dgi	bool	0x1	Enable sending multichip DGI messages in a separate thread. This parameter is valid only if enable-multichip-operation is set.
gic_iri.non-ARE-core-count	int	0x8	Maximum number of non-ARE cores; normally used to pass the cluster-level NUM_CORES parameter to the top-level redistributor.
gic_iri.outer-cacheability-support	bool	0x0	Allow configuration of outer cachability attributes in ITS and Redistributor
gic_iri.output_attributes	string	"ExtendedID[62:55]=MPAM_PMG, ExtendedID[54:39]=MPAM_PARTID, ExtendedID[38]=MPAM_NS"	User-defined transform to be applied to bus attributes like MasterID, ExtendedID or UserFlags. Currently, only works for MPAM Attributes encoding into bus attributes.
gic_iri.print-memory-map	bool	0x0	Print memory map to stdout
gic_iri.priority-bits	int	0x5	Number of implemented priority bits
gic_iri.processor-numbers	string	""	Specify processor numbers (as appears in GICR_TYPER) in the form 0.0.0.0=0,0.0.0.1=1 etc.) If not specified, will number processors starting at 0.
gic_iri.redistributor-threaded-sync	bool	0x1	Enable execution of redistributor delayed transactions in a separate thread which is sometimes required for cosimulation
gic_iri.reg-base	int	0x2c010000	Base for decoding GICv3 registers.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-205 Parameters for gic_iri (continued)

Name	Type	Default value	Description
gic_iri.reg-base-per-redistributor	string	""	Base address for each redistributor in the form '0.0.0.0=0x2c010000, 0.0.0.1=0x2c020000'. All redistributors must be specified and this overrides the reg-base parameter (except that reg-base will still be used for the top-level redistributor). If reg-base-per-redistributor-file is specified, this parameter is ignored.
gic_iri.reg-base-per-redistributor-file	string	""	Path to file containing the base address for each redistributor in the form '0.0.0.0=0x2c010000, 0.0.0.1=0x2c020000'. All redistributors must be specified and this overrides the reg-base parameter (except that reg-base will still be used for the top-level redistributor). If this parameter is specified, reg-base-per-redistributor parameter will be ignored even when it is given.
gic_iri.report-MSI-error-via-statusr	int	0x0	Report MSI error via GITS_STATUSR. (0:unsupported, 1:report by GITS_STATUSR, 2:report by GITS_STATUSR and interrupt as well).
gic_iri.sgi-range-selector-support	bool	0x0	Device has support for the Range Selector feature for SGI
gic_iri.single-set-support	bool	0x0	When true, forces redistributors to recall interrupts with a clear rather than issue a second Set command
gic_iri.supports-shareability	bool	0x1	Device supports shareability attributes on outgoing memory bus (i.e. is modelling an ACElite port rather than an AXI4 port).
gic_iri.trace-speculative-lpi-property-update	bool	0x0	Trace LPI propery updates performed on speculative accesses (useful for debuging LPI)
gic_iri.virtual-lpi-support	bool	0x0	GICv4 Virtual LPIs and Direct injection of Virtual LPIs supported
gic_iri.virtual-priority-bits	int	0x5	Number of implemented virtual priority bits
gic_iri.wakeup-on-reset	bool	0x0	Go against specification and start redistributors in woken-up state at reset. This allows software that was written for previous versions of the GICv3 specification to work correctly. This should not be used for production code or when the distributor is used separately from the core fast model.

3.4.45 ARMCortexX1CT

ARMCortexX1CT CPU component. This model is written in C++ and models version r0p0 of the RTL.

ARMCortexX1CT contains the following CADI targets:

- ARM_Cortex-X1
- Cluster_ARM_Cortex-X1
- PVCache
- TlbCadi

ARMCortexX1CT contains the following MTI components:

- *4.42 ARM_Cortex-X1 - trace* on page 4-2196
- *4.48 ARMv8Cluster - trace* on page 4-2401
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402
- *4.68 DSU - trace* on page 4-2463
- *4.73 GICv3CPUInterfaceDecoder - trace* on page 4-2489
- *4.104 PVBusLogger - trace* on page 4-2756

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.108 PVCache - trace* on page 4-2761
- *4.114 TLB - trace* on page 4-2834

About ARMCortexX1CT

The model supports the following features:

- DynamIQ (FCM) system registers.
- Per-core L2 cache.
- A P-Channel for the cluster and for each core.
- Optional peripheral port.
- L3 cache partition.
- Per-core clock.
- `BROADCASTPERSIST` pin.

The model does not support the following features:

- `BROADCASTCACHEMAINTPOU` pin.
- `COREINSTRRET`, `COREINSTRRUN`, or `nPMBIRQ` signals.
- DynamIQ features that are irrelevant to the programmers' view simulation, for example:
  - Automatic CPU retention mode.
  - Level-3 Cache RAM retention.
- 256-bit wide output transactions.
- ECC and parity schemes are hardware-specific so are not supported.
- Self-test features (MBIST).
- Latency configuration.
- Snoop filtering.
- Debug power management signals `DBGPWRDUP` and `DBGRSTREQ` are not implemented, but `DBGPWRUPREQ` and `DBGNOPWRDWN` are implemented.
- Cache stashing capability.
- Each thread has its own L1Cache and L2Cache instead of a common cache that is shared by all threads of the core.
- Per-thread parameters are not yet supported but signals are implemented.

This component has a variable number of cores per cluster, specified using the `NUM_CORES` parameter. The following components also exist, with a fixed number of cores per cluster:

- ARMCortexX1x1CT.
- ARMCortexX1x2CT.
- ARMCortexX1x3CT.
- ARMCortexX1x4CT.

The per-core parameters are preceded by `cpu`*n*`.`, where *n* identifies the core (0-3).

Table 3-206 Ports

Name	Protocol	Type	Description
`AENDMP`	*Value_64* on page 2-85	Slave	Port to obtain end address of valid peripheral address range (exclusive).
`ASTARTMP`	*Value_64* on page 2-85	Slave	Port to obtain start address of valid peripheral address range (inclusive).
`CNTHPIRQ[4]`	*Signal* on page 2-85	Master	Timer signals to SOC.
`CNTHVIRQ[4]`	*Signal* on page 2-85	Master	Timer signals to SOC.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-206 Ports (continued)

Name	Protocol	Type	Description
CNTPNSIRQ[4]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTPSIRQ[4]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTVIRQ[4]	*Signal* on page 2-85	Master	Timer signals to SOC.
acp_s	PVBus	Slave	AXI ACP slave port.
broadcastatomic	*Signal* on page 2-85	Slave	CHI defined pins.
broadcastcachemaint	*Signal* on page 2-85	Slave	ACE defined pins.
broadcastouter	*Signal* on page 2-85	Slave	ACE defined pins.
broadcastpersist	*Signal* on page 2-85	Slave	CHI defined pins.
cfgend[4]	*Signal* on page 2-85	Slave	This signal if for EE bit initialisation.
cfgsdisable	*Signal* on page 2-85	Slave	This signal disables write access to some secure Interrupt Controller registers.
cfgte[4]	*Signal* on page 2-85	Slave	This signal provides default exception handling state.
clk_in	ClockSignal	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the cluster level components run e.g cluster level timers, caches and pmu.
clusterid	*Value* on page 2-85	Slave	The port reads the value in CPU ID register field, bits[11:8] of the MPIDR.
clusterpmuirq	*Signal* on page 2-85	Master	DynamIQ pmu irq
cntvalueb	*CounterInterface* on page 2-84	Slave	Interface to SoC level counter module.
commirq[4]	*Signal* on page 2-85	Master	Interrupt signal from debug communications channel.
core_clk_in[4]	ClockSignal	Slave	The clock signal connected to the core_clk_in port is used to determine the rate at which each core executes instructions.
cpuporeset[4]	*Signal* on page 2-85	Slave	CPU power on reset. Initializes all the processor logic, including debug logic.
cryptodisable[4]	*Signal* on page 2-85	Slave	Disable cryptography extensions after reset.
cti[4]	*v8EmbeddedCrossTrigger_controlprotocol* on page 2-84	Master	Cross trigger matrix port.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-206 Ports (continued)

Name	Protocol	Type	Description
ctidbgirq[4]	*Signal* on page 2-85	Master	Cross Trigger Interface (CTI) interrupt trigger output.
dbgen[4]	*Signal* on page 2-85	Slave	External debug interface.
dbgnopwrdwn[4]	*Signal* on page 2-85	Master	No power-down request.
dbgpwrupreq[4]	*Signal* on page 2-85	Master	Debug power up request.
dev_debug_s	PVBus	Slave	External debug interface.
event	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE.
fiq[4]	*Signal* on page 2-85	Slave	This signal drives the CPUs fast-interrupt handling.
gicv3_redistributor_s[4]	*GICv3Comms* on page 2-84	Slave	GICv3 AXI-stream port.
irq[4]	*Signal* on page 2-85	Slave	This signal drives the CPUs interrupt handling.
memorymapped_debug_s	PVBus	Slave	External debug interface.
niden[4]	*Signal* on page 2-85	Slave	External debug interface.
pchannel_cluster	*PChannel* on page 2-80	Slave	PChannel for cluster.
pchannel_core[4]	*PChannel* on page 2-80	Slave	PChannels for cores
pmuirq[4]	*Signal* on page 2-85	Master	Interrupt signal from performance monitoring unit.
presetdbg	*Signal* on page 2-85	Slave	Initialize the shared debug APB, Cross Trigger Interface (CTI), and Cross Trigger Matrix (CTM) logic.
pvbus_m0	PVBus	Master	The core will generate bus requests on this port.
pvbus_periph_m	PVBus	Master	The core can generate peripheral bus request on this port.
rei[4]	*Signal* on page 2-85	Slave	Per core RAM Error Interrupt.
reset[4]	*Signal* on page 2-85	Slave	Raising this signal will put the core into reset mode.
rvbaraddr[4]	*Value_64* on page 2-85	Slave	Reset vector base address.
sci_m	SystemCoherencyInterface	Master	System coherency interface port, which is used to take the whole cluster into/out-of coherency domain

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-206 Ports (continued)

Name	Protocol	Type	Description
sei[4]	*Signal* on page 2-85	Slave	Per core System Error physical pins.
spiden[4]	*Signal* on page 2-85	Slave	External debug interface.
spniden[4]	*Signal* on page 2-85	Slave	External debug interface.
sporeset	*Signal* on page 2-85	Slave	A single cluster-wide power on reset signal for all resettable registers in DynamIQ.
ticks[4]	*InstructionCount* on page 2-84	Master	This port should be connected to one of the two ticks ports on a 'visualisation' component, in order to display a running instruction count.
vcpumntirq[4]	*Signal* on page 2-85	Master	Interrupt signal for virtual CPU maintenance IRQ.
vfiq[4]	*Signal* on page 2-85	Slave	Virtualised FIQ.
vinithi[4]	*Signal* on page 2-85	Slave	This signal controls of the location of the exception vectors at reset.
virq[4]	*Signal* on page 2-85	Slave	Virtualised IRQ.
virtio_s	PVBus	Slave	The virtio coherent port, hooks directly into the L2 system and becomes coherent (assuming attributes are set correctly).
vsei[4]	*Signal* on page 2-85	Slave	Per core virtual System Error physical pins.

Table 3-207 Parameters for Cluster_ARM_Cortex-X1

Name	Type	Default value	Description
BROADCASTATOMIC	bool	0x1	Enable broadcasting of atomic operation. The broadcastatomic signal will override this value if used
BROADCASTCACHEMAINT	bool	0x1	Enable broadcasting of cache maintenance operations to downstream caches. The broadcastcachemaint signal will override this value if used.
BROADCASTOUTER	bool	0x1	Enable broadcasting of Outer Shareable transactions. The broadcastouter signal will override this value if used.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-207 Parameters for Cluster_ARM_Cortex-X1 (continued)

Name	Type	Default value	Description
BROADCASTPERSIST	bool	0x1	Enable broadcasting of cache clean to the point of persistence operations. The broadcastpersist signal will override this value if used
CLUSTER_ID	int	0x0	Processor cluster ID value
GICDISABLE	bool	0x1	Disable the new style GICv3 CPU interface in each core model. Should be left enabled unless the platform contains a GICv3 distributor.
cpi_div	int	0x1	Divider for calculating CPI (Cycles Per Instruction)
cpi_mul	int	0x1	Multiplier for calculating CPI (Cycles Per Instruction)
dcache-hit_latency	int	0x0	L1 D-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
dcache-maintenance_latency	int	0x0	L1 D-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
dcache-miss_latency	int	0x0	L1 D-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
dcache-prefetch_enabled	bool	0x0	Enable simulation of data cache prefetching. This is only used when dcache-state_modelled=true
dcache-read_access_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per access (of size dcache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
dcache-read_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per byte accessed.dcache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
dcache-size	int	0x10000	L1 D-Cache size in bytes.
dcache-snoop_data_transfer_latency	int	0x0	L1 D-Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-207 Parameters for Cluster_ARM_Cortex-X1 (continued)

Name	Type	Default value	Description
dcache-state_modelled	bool	0x0	Set whether D-cache has stateful implementation
dcache-write_access_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per access (of size dcache-write_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-write_latency is set. This is only used when dcache-state_modelled=true.
dcache-write_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per byte accessed. dcache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
default_opmode	int	0x4	Operating mode of DynamIQ coming out of reset. 0: SFONLY ON, 1: 1/4 CACHE ON, 2: 1/2 CACHE ON, 3: 3/4 CACHE ON, 4: FULL CACHE ON
diagnostics	bool	0x0	Enable DynamIQ diagnostic messages
enable_simulation_performance_optimizations	bool	0x1	With this option enabled, the model will run more quickly, but be less accurate to exact CPU behavior. The model will still be functionally accurate for software, but may increase differences seen between hardware behavior and model behavior for certain workloads (it changes the micro-architectural value of stage12_tlb_size parameter to 1024).
ext_abort_device_read_is_sync	bool	0x0	Synchronous reporting of device-nGnRE read external aborts
ext_abort_device_write_is_sync	bool	0x0	Synchronous reporting of device-nGnRE write external aborts
ext_abort_so_read_is_sync	bool	0x0	Synchronous reporting of device-nGnRnE read external aborts
ext_abort_so_write_is_sync	bool	0x0	Synchronous reporting of device-nGnRnE write external aborts
has_coherent_icache	bool	0x0	Whether icache invalidation to the point of unification is required for instruction to data coherence. true - Invalidate operations not required
has_dot_product	int	0x2	Possible values of this parameter are: - 1, feature not implemented. - 2, feature is implemented.
has_peripheral_port	bool	0x0	If true, additional AXI peripheral port is configured.
icache-hit_latency	int	0x0	L1 I-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when icache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-207 Parameters for Cluster_ARM_Cortex-X1 (continued)

Name	Type	Default value	Description
icache-maintenance_latency	int	0x0	L1 I-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when icache-state_modelled=true.
icache-miss_latency	int	0x0	L1 I-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when icache-state_modelled=true.
icache-prefetch_enabled	bool	0x0	Enable simulation of instruction cache prefetching. This is only used when icache-state_modelled=true.
icache-read_access_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per access (of size icache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if icache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when icache-state_modelled=true.
icache-read_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per byte accessed.icache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when icache-state_modelled=true.
icache-size	int	0x10000	L1 I-Cache size in bytes.
icache-state_modelled	bool	0x0	Set whether I-cache has stateful implementation
l3cache-hit_latency	int	0x0	L3 Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
l3cache-maintenance_latency	int	0x0	L3 Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
l3cache-miss_latency	int	0x0	L3 Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-207 Parameters for Cluster_ARM_Cortex-X1 (continued)

Name	Type	Default value	Description
l3cache-read_access_latency	int	0x0	L3 Cache timing annotation latency for read accesses given in ticks per access (of size l3cache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l3cache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
l3cache-read_latency	int	0x0	L3 Cache timing annotation latency for read accesses given in ticks per byte accessed.l3cache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
l3cache-size	int	0x100000	L3 Cache size in bytes.
l3cache-snoop_data_transfer_latency	int	0x0	L3 Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
l3cache-snoop_issue_latency	int	0x0	L3 Cache timing annotation latency for snoop accesses issued by this cache in total ticks. This is only used when dcache-state_modelled=true.
l3cache-write_access_latency	int	0x0	L3 Cache timing annotation latency for write accesses given in ticks per access (of size l3cache-write_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l3cache-write_latency is set. This is only used when dcache-state_modelled=true.
l3cache-write_latency	int	0x0	L3 Cache timing annotation latency for write accesses given in ticks per byte accessed. l3cache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
pchannel_treat_simreset_as_poreset	bool	0x0	Register core as ON state to cluster with simulation reset.
periph_address_end	int	0x0	End address for peripheral port address range exclusive(corresponds to AENDMP input signal).
periph_address_start	int	0x0	Start address for peripheral port address range inclusive(corresponds to ASTARTMP input signal).
ptw_latency	int	0x0	Page table walker latency for TA (Timing Annotation), expressed in simulation ticks

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-207 Parameters for Cluster_ARM_Cortex-X1 (continued)

Name	Type	Default value	Description
tlb_latency	int	0x0	TLB latency for TA (Timing Annotation), expressed in simulation ticks
treat-dcache-cmos-to-pou-as-nop	bool	0x0	Whether dcache invalidation to the point of unification is required for instruction to data coherence. true - Invalidate operations not required
walk_cache_latency	int	0x0	Walk cache latency for TA (Timing Annotation), expressed in simulation ticks

Table 3-208 Parameters for ARM_Cortex-X1

Name	Type	Default value	Description
cpu0.CFGEND	bool	0x0	Endianness configuration at reset. 0, little endian. 1, big endian.
cpu0.CFGTE	bool	0x0	Instruction set state when resetting into AArch32. 0, A32. 1, T32.
cpu0.CRYPTODISABLE	bool	0x0	Disable cryptographic features.
cpu0.RVBARADDR	int	0x0	Value of RVBAR_ELx register.
cpu0.VINITHI	bool	0x0	Reset value of SCTLR.V.
cpu0.enable_trace_special_hlt_imm16	bool	0x0	Enable usage of parameter trace_special_hlt_imm16
cpu0.l2cache-hit_latency	int	0x0	L2 Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
cpu0.l2cache-maintenance_latency	int	0x0	L2 Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
cpu0.l2cache-miss_latency	int	0x0	L2 Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
cpu0.l2cache-read_access_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-208 Parameters for ARM_Cortex-X1 (continued)

Name	Type	Default value	Description
cpu0.l2cache-read_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per byte accessed.l2cache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
cpu0.l2cache-size	int	0x80000	L2 Cache size in bytes.
cpu0.l2cache-snoop_data_transfer_latency	int	0x0	L2 Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
cpu0.l2cache-snoop_issue_latency	int	0x0	L2 Cache timing annotation latency for snoop accesses issued by this cache in total ticks. This is only used when dcache-state_modelled=true.
cpu0.l2cache-write_access_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-write_latency is set. This is only used when dcache-state_modelled=true.
cpu0.l2cache-write_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per byte accessed. l2cache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
cpu0.max_code_cache_mb	int	0x100	Maximum size of the simulation code cache (MiB). For platforms with more than 2 cores this limit will be scaled down. (e.g 1/8 for 16 or more cores)
cpu0.min_sync_level	int	0x0	Force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)
cpu0.semihosting-A32_HLT	int	0xf000	A32 HLT number for semihosting calls.
cpu0.semihosting-A64_HLT	int	0xf000	A64 HLT number for semihosting calls.
cpu0.semihosting-ARM_SVC	int	0x123456	A32 SVC number for semihosting calls.
cpu0.semihosting-T32_HLT	int	0x3c	T32 HLT number for semihosting calls.
cpu0.semihosting-Thumb_SVC	int	0xab	T32 SVC number for semihosting calls.
cpu0.semihosting-cmd_line	string	""	Command line available to semihosting calls.
cpu0.semihosting-cwd	string	""	Base directory for semihosting file access.
cpu0.semihosting-enable	bool	0x1	Enable semihosting SVC/HLT traps.
cpu0.semihosting-heap_base	int	0x0	Virtual address of heap base.
cpu0.semihosting-heap_limit	int	0xf000000	Virtual address of top of heap.
cpu0.semihosting-stack_base	int	0x10000000	Virtual address of base of descending stack.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-208 Parameters for ARM_Cortex-X1 (continued)

Name	Type	Default value	Description
cpu0.semihosting-stack_limit	int	0xf000000	Virtual address of stack limit.
cpu0.trace_special_hlt_imm16	int	0xf000	For this HLT number, IF enable_trace_special_hlt_imm16=true, skip performing usual HLT execution but call MTI trace if registered
cpu0.vfp-enable_at_reset	bool	0x0	Enable VFP in CPACR, CPPWR, NSACR at reset. Warning: Arm recommends going through the implementation's suggested VFP power-up sequence!

3.4.46 ARMNeoverseE1CT

ARMNeoverseE1CT CPU component. This model is written in C++.

ARMNeoverseE1CT contains the following CADI targets:

- ARM_Neoverse-E1
- Cluster_ARM_Neoverse-E1
- PVCache
- TlbCadi

ARMNeoverseE1CT contains the following MTI components:

- *4.44 ARM_Neoverse-E1 - trace* on page 4-2273
- *4.48 ARMv8Cluster - trace* on page 4-2401
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402
- *4.68 DSU - trace* on page 4-2463
- *4.73 GICv3CPUInterfaceDecoder - trace* on page 4-2489
- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.108 PVCache - trace* on page 4-2761
- *4.114 TLB - trace* on page 4-2834

About ARMNeoverseE1CT

The model supports the following features:

- DynamIQ (FCM) system registers.
- Per-core L2 cache.
- A P-Channel for the cluster and for each core.
- Optional peripheral port.
- L3 cache partition.
- Per-core clock.
- BROADCASTPERSIST pin.

The model does not support the following features:

- BROADCASTCACHEMAINTPOU pin.
- COREINSTRRET, COREINSTRRUN, or nPMBIRQ signals.
- DynamIQ features that are irrelevant to the programmers' view simulation, for example:
  - Automatic CPU retention mode.
  - Level-3 Cache RAM retention.
- 256-bit wide output transactions.
- ECC and parity schemes are hardware-specific so are not supported.
- Self-test features (MBIST).
- Latency configuration.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

- Snoop filtering.
- Debug power management signals DBGPWRDUP and DBGRSTREQ are not implemented, but DBGPWRUPREQ and DBGNOPWRDWN are implemented.
- Cache stashing capability.
- Each thread has its own L1Cache and L2Cache instead of a common cache that is shared by all threads of the core.
- Per-thread parameters are not yet supported but signals are implemented.

This model has a variable number of cores per cluster, specified using the NUM_CORES parameter. The following models also exist, with a fixed number of cores per cluster:

- ARMNeoverseE1x1CT.
- ARMNeoverseE1x2CT.
- ARMNeoverseE1x3CT.
- ARMNeoverseE1x4CT.

The per-core parameters are preceded by cpu*n*., where *n* identifies the core (0-3).

Table 3-209 Ports

Name	Protocol	Type	Description
AENDMP	*Value_64* on page 2-85	Slave	Port to obtain end address of valid peripheral address range (exclusive).
ASTARTMP	*Value_64* on page 2-85	Slave	Port to obtain start address of valid peripheral address range (inclusive).
CNTHPIRQ[16]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTHVIRQ[16]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTPNSIRQ[16]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTPSIRQ[16]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTVIRQ[16]	*Signal* on page 2-85	Master	Timer signals to SOC.
acp_s	PVBus	Slave	AXI ACP slave port
broadcastatomic	*Signal* on page 2-85	Slave	CHI defined pins.
broadcastcachemaint	*Signal* on page 2-85	Slave	ACE defined pins.
broadcastouter	*Signal* on page 2-85	Slave	ACE defined pins.
broadcastpersist	*Signal* on page 2-85	Slave	CHI defined pins.
cfgend[16]	*Signal* on page 2-85	Slave	This signal if for EE bit initialisation.
cfgsdisable	*Signal* on page 2-85	Slave	This signal disables write access to some secure Interrupt Controller registers.
cfgte[16]	*Signal* on page 2-85	Slave	This signal provides default exception handling state.
clk_in	ClockSignal	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the core executes instructions.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-209 Ports (continued)

Name	Protocol	Type	Description
clusterid	*Value* on page 2-85	Slave	The port reads the value in CPU ID register field, bits[11:8] of the MPIDR.
clusterpmuirq	*Signal* on page 2-85	Master	DynamIQ pmu irq
cntvalueb	*CounterInterface* on page 2-84	Slave	Interface to SoC level counter module.
commirq[16]	*Signal* on page 2-85	Master	Interrupt signal from debug communications channel.
core_clk_in[8]	ClockSignal	Slave	The clock signal connected to the core_clk_in port is used to determine the rate at which each core executes instructions.
cp15sdisable[16]	*Signal* on page 2-85	Slave	This signal disables write access to some system control processor registers.
cpuporeset[8]	*Signal* on page 2-85	Slave	Power on reset. Initializes all the processor logic, including debug logic.
cryptodisable[16]	*Signal* on page 2-85	Slave	Disable cryptography extensions after reset.
cti[16]	*v8EmbeddedCrossTrigger_controlprotocol* on page 2-84	Master	Cross trigger matrix port.
ctidbgirq[16]	*Signal* on page 2-85	Master	Cross Trigger Interface (CTI) interrupt trigger output.
dbgen[16]	*Signal* on page 2-85	Slave	External debug interface.
dbgnopwrdwn[16]	*Signal* on page 2-85	Master	These signals relate to core power down.
dbgpwrupreq[16]	*Signal* on page 2-85	Master	Debug power up request.
dev_debug_s	PVBus	Slave	External debug interface.
event	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE.
fiq[16]	*Signal* on page 2-85	Slave	This signal drives the CPUs fast-interrupt handling.
gicv3_redistributor_s[16]	*GICv3Comms* on page 2-84	Slave	GICv3 AXI-stream port per thread.
irq[16]	*Signal* on page 2-85	Slave	This signal drives the CPUs interrupt handling.
memorymapped_debug_s	PVBus	Slave	External debug interface.
niden[16]	*Signal* on page 2-85	Slave	External debug interface.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-209 Ports (continued)

Name	Protocol	Type	Description
pchannel_cluster	*PChannel* on page 2-80	Slave	PChannel for cluster.
pchannel_core[8]	*PChannel* on page 2-80	Slave	PChannels for cores
pmuirq[16]	*Signal* on page 2-85	Master	Interrupt signal from performance monitoring unit.
presetdbg	*Signal* on page 2-85	Slave	Initialize the shared debug APB, Cross Trigger Interface (CTI), and Cross Trigger Matrix (CTM) logic.
pvbus_m0	PVBus	Master	The core will generate bus requests on this port.
pvbus_periph_m	PVBus	Master	The core can generate peripheral bus request on this port.
rei[16]	*Signal* on page 2-85	Slave	Per core RAM Error Interrupt.
reset[8]	*Signal* on page 2-85	Slave	Raising this signal will put the core into reset mode.
rvbaraddr[16]	*Value_64* on page 2-85	Slave	Reset vector base address.
sci_m	SystemCoherencyInterface	Master	System coherency interface port, which is used to take the whole cluster into/out-of coherency domain
sei[16]	*Signal* on page 2-85	Slave	Per core virtual System Error physical pins.
spiden[16]	*Signal* on page 2-85	Slave	External debug interface.
spniden[16]	*Signal* on page 2-85	Slave	External debug interface.
sporeset	*Signal* on page 2-85	Slave	A single cluster-wide power on reset signal for all resettable registers in DynamIQ.
ticks[16]	*InstructionCount* on page 2-84	Master	This port should be connected to one of the two ticks ports on a 'visualisation' component, in order to display a running instruction count.
vcpumntirq[16]	*Signal* on page 2-85	Master	Interrupt signal for virtual CPU maintenance IRQ.
vfiq[16]	*Signal* on page 2-85	Slave	Virtualised FIQ.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-209 Ports (continued)

Name	Protocol	Type	Description
vinithi[16]	*Signal* on page 2-85	Slave	This signal controls of the location of the exception vectors at reset.
virq[16]	*Signal* on page 2-85	Slave	Virtualised IRQ.
virtio_s	PVBus	Slave	The virtio coherent port, hooks directly into the L2 system and becomes coherent (assuming attributes are set correctly).
vsei[16]	*Signal* on page 2-85	Slave	Per core virtual System Error physical pins.

Table 3-210 Parameters for Cluster_ARM_Neoverse-E1

Name	Type	Default value	Description
BROADCASTATOMIC	bool	0x1	Enable broadcasting of atomic operation. The broadcastatomic signal will override this value if used
BROADCASTCACHEMAINT	bool	0x1	Enable broadcasting of cache maintenance operations to downstream caches. The broadcastcachemaint signal will override this value if used.
BROADCASTOUTER	bool	0x1	Enable broadcasting of Outer Shareable transactions. The broadcastouter signal will override this value if used.
BROADCASTPERSIST	bool	0x1	Enable broadcasting of cache clean to the point of persistence operations. The broadcastpersist signal will override this value if used
CLUSTER_ID	int	0x0	Processor cluster ID value
GICDISABLE	bool	0x1	Disable the new style GICv3 CPU interface in each core model. Should be left enabled unless the platform contains a GICv3 distributor.
cluster_patch_level	int	0x0	Cosmetic change to patch number in CLUSTERIDR. Corresponds to the Y in rXpY.
cluster_revision_number	int	0x0	Cosmetic change to revision number in CLUSTERIDR, Corresponds to the X in rXpY.
cpi_div	int	0x1	Divider for calculating CPI (Cycles Per Instruction)
cpi_mul	int	0x1	Multiplier for calculating CPI (Cycles Per Instruction)
dcache-hit_latency	int	0x0	L1 D-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-210 Parameters for Cluster_ARM_Neoverse-E1 (continued)

Name	Type	Default value	Description
dcache-maintenance_latency	int	0x0	L1 D-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
dcache-miss_latency	int	0x0	L1 D-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
dcache-prefetch_enabled	bool	0x0	Enable simulation of data cache prefetching. This is only used when dcache-state_modelled=true
dcache-read_access_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per access (of size dcache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
dcache-read_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per byte accessed.dcache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
dcache-size	int	0x10000	L1 D-Cache size in bytes.
dcache-snoop_data_transfer_latency	int	0x0	L1 D-Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
dcache-state_modelled	bool	0x0	Set whether D-cache has stateful implementation
dcache-write_access_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per access (of size dcache-write_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-write_latency is set. This is only used when dcache-state_modelled=true.
dcache-write_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per byte accessed. dcache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-210 Parameters for Cluster_ARM_Neoverse-E1 (continued)

Name	Type	Default value	Description
default_opmode	int	0x4	Operating mode of DynamIQ coming out of reset. 0: SFONLY ON, 1: 1/4 CACHE ON, 2: 1/2 CACHE ON, 3: 3/4 CACHE ON, 4: FULL CACHE ON
diagnostics	bool	0x0	Enable DynamIQ diagnostic messages
enable_simulation_performance_optimizations	bool	0x1	With this option enabled, the model will run more quickly, but be less accurate to exact CPU behavior. The model will still be functionally accurate for software, but may increase differences seen between hardware behavior and model behavior for certain workloads (it changes the micro-architectural value of stage12_tlb_size parameter to 1024).
has_dot_product	int	0x2	Possible values of this parameter are: - 1, feature not implemented. - 2, feature is implemented.
has_peripheral_port	bool	0x0	If true, additional AXI peripheral port is configured.
icache-hit_latency	int	0x0	L1 I-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when icache-state_modelled=true.
icache-maintenance_latency	int	0x0	L1 I-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when icache-state_modelled=true.
icache-miss_latency	int	0x0	L1 I-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when icache-state_modelled=true.
icache-prefetch_enabled	bool	0x0	Enable simulation of instruction cache prefetching. This is only used when icache-state_modelled=true.
icache-read_access_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per access (of size icache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if icache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when icache-state_modelled=true.
icache-read_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per byte accessed.icache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when icache-state_modelled=true.
icache-size	int	0x10000	L1 I-Cache size in bytes.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-210 Parameters for Cluster_ARM_Neoverse-E1 (continued)

Name	Type	Default value	Description
icache-state_modelled	bool	0x0	Set whether I-cache has stateful implementation
l3cache-hit_latency	int	0x0	L3 Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
l3cache-maintenance_latency	int	0x0	L3 Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
l3cache-miss_latency	int	0x0	L3 Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
l3cache-read_access_latency	int	0x0	L3 Cache timing annotation latency for read accesses given in ticks per access (of size l3cache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l3cache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
l3cache-read_latency	int	0x0	L3 Cache timing annotation latency for read accesses given in ticks per byte accessed.l3cache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
l3cache-size	int	0x400000	L3 Cache size in bytes.
l3cache-snoop_data_transfer_latency	int	0x0	L3 Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
l3cache-snoop_issue_latency	int	0x0	L3 Cache timing annotation latency for snoop accesses issued by this cache in total ticks. This is only used when dcache-state_modelled=true.
l3cache-write_access_latency	int	0x0	L3 Cache timing annotation latency for write accesses given in ticks per access (of size l3cache-write_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l3cache-write_latency is set. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-210 Parameters for Cluster_ARM_Neoverse-E1 (continued)

Name	Type	Default value	Description
l3cache-write_latency	int	0x0	L3 Cache timing annotation latency for write accesses given in ticks per byte accessed. l3cache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
pchannel_treat_simreset_as_poreset	bool	0x0	Register core as ON state to cluster with simulation reset.
periph_address_end	int	0x0	End address for peripheral port address range exclusive(corresponds to AENDMP input signal).
periph_address_start	int	0x0	Start address for peripheral port address range inclusive(corresponds to ASTARTMP input signal).
ptw_latency	int	0x0	Page table walker latency for TA (Timing Annotation), expressed in simulation ticks
tlb_latency	int	0x0	TLB latency for TA (Timing Annotation), expressed in simulation ticks
walk_cache_latency	int	0x0	Walk cache latency for TA (Timing Annotation), expressed in simulation ticks

Table 3-211 Parameters for ARM_Neoverse-E1

Name	Type	Default value	Description
cpu0.thread0.CRYPTODISABLE	bool	0x0	Disable cryptographic features.
cpu0.thread0.enable_single_thread_at_reset	bool	0x0	Enable single thread after reset and keep other thread in reset
cpu0.thread0.enable_trace_special_hlt_imm16	bool	0x0	Enable usage of parameter trace_special_hlt_imm16
cpu0.thread0.l2cache-hit_latency	int	0x0	L2 Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
cpu0.thread0.l2cache-maintenance_latency	int	0x0	L2 Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
cpu0.thread0.l2cache-miss_latency	int	0x0	L2 Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-211 Parameters for ARM_Neoverse-E1 (continued)

Name	Type	Default value	Description
cpu0.thread0.l2cache-read_access_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
cpu0.thread0.l2cache-read_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per byte accessed.l2cache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
cpu0.thread0.l2cache-size	int	0x40000	L2 Cache size in bytes.
cpu0.thread0.l2cache-snoop_data_transfer_latency	int	0x0	L2 Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
cpu0.thread0.l2cache-snoop_issue_latency	int	0x0	L2 Cache timing annotation latency for snoop accesses issued by this cache in total ticks. This is only used when dcache-state_modelled=true.
cpu0.thread0.l2cache-write_access_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-write_latency is set. This is only used when dcache-state_modelled=true.
cpu0.thread0.l2cache-write_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per byte accessed. l2cache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
cpu0.thread0.max_code_cache_mb	int	0x100	Maximum size of the simulation code cache (MiB). For platforms with more than 2 cores this limit will be scaled down. (e.g 1/8 for 16 or more cores)
cpu0.thread0.min_sync_level	int	0x0	Force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-211 Parameters for ARM_Neoverse-E1 (continued)

Name	Type	Default value	Description
cpu0.thread0.semihosting-A32_HLT	int	0xf000	A32 HLT number for semihosting calls.
cpu0.thread0.semihosting-A64_HLT	int	0xf000	A64 HLT number for semihosting calls.
cpu0.thread0.semihosting-ARM_SVC	int	0x123456	A32 SVC number for semihosting calls.
cpu0.thread0.semihosting-T32_HLT	int	0x3c	T32 HLT number for semihosting calls.
cpu0.thread0.semihosting-Thumb_SVC	int	0xab	T32 SVC number for semihosting calls.
cpu0.thread0.semihosting-cmd_line	string	""	Command line available to semihosting calls.
cpu0.thread0.semihosting-cwd	string	""	Base directory for semihosting file access.
cpu0.thread0.semihosting-enable	bool	0x1	Enable semihosting SVC/HLT traps.
cpu0.thread0.semihosting-heap_base	int	0x0	Virtual address of heap base.
cpu0.thread0.semihosting-heap_limit	int	0xf000000	Virtual address of top of heap.
cpu0.thread0.semihosting-stack_base	int	0x10000000	Virtual address of base of descending stack.
cpu0.thread0.semihosting-stack_limit	int	0xf000000	Virtual address of stack limit.
cpu0.thread0.trace_special_hlt_imm16	int	0xf000	For this HLT number, IF enable_trace_special_hlt_imm16=true, skip performing usual HLT execution but call MTI trace if registered
cpu0.thread0.vfp-enable_at_reset	bool	0x0	Enable VFP in CPACR, CPPWR, NSACR at reset. Warning: Arm recommends going through the implementation's suggested VFP power-up sequence!
cpu0.thread0.vfp-present	bool	0x1	Set whether the model has VFP support

3.4.47 ARMNeoverseN1CT

ARMNeoverseN1CT CPU component. This model is written in C++ and models version r0p0 of the RTL.

ARMNeoverseN1CT contains the following CADI targets:

- ARM_Neoverse-N1
- Cluster_ARM_Neoverse-N1
- PVCache
- TlbCadi

ARMNeoverseN1CT contains the following MTI components:

- *4.45 ARM_Neoverse-N1 - trace* on page 4-2311
- *4.48 ARMv8Cluster - trace* on page 4-2401
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402
- *4.68 DSU - trace* on page 4-2463
- *4.73 GICv3CPUInterfaceDecoder - trace* on page 4-2489
- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.108 PVCache - trace* on page 4-2761
- *4.114 TLB - trace* on page 4-2834

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

About ARMNeoverseN1CT

The model supports the following features:

- DynamIQ (FCM) system registers.
- Per-core L2 cache.
- A P-Channel for the cluster and for each core.
- Optional peripheral port.
- L3 cache partition.
- Per-core clock.
- BROADCASTPERSIST pin.

The model does not support the following features:

- BROADCASTCACHEMAINTPOU pin.
- COREINSTRRET, COREINSTRRUN, or nPMBIRQ signals.
- DynamIQ features that are irrelevant to the programmers' view simulation, for example:
  — Automatic CPU retention mode.
  — Level-3 Cache RAM retention.
- 256-bit wide output transactions.
- ECC and parity schemes are hardware-specific so are not supported.
- Self-test features (MBIST).
- Latency configuration.
- Snoop filtering.
- Debug power management signals DBGPWRDUP and DBGRSTREQ are not implemented, but DBGPWRUPREQ and DBGNOPWRDWN are implemented.
- Cache stashing capability.

This model has a variable number of cores per cluster, specified using the NUM_CORES parameter. The following models also exist, with a fixed number of cores per cluster:

- ARMNeoverseN1x1CT.
- ARMNeoverseN1x2CT.
- ARMNeoverseN1x3CT.
- ARMNeoverseN1x4CT.

The per-core parameters are preceded by cpu*n*., where *n* identifies the core (0-3).

Table 3-212 Ports

Name	Protocol	Type	Description
AENDMP	*Value_64* on page 2-85	Slave	Port to obtain end address of valid peripheral address range (exclusive).
ASTARTMP	*Value_64* on page 2-85	Slave	Port to obtain start address of valid peripheral address range (inclusive).
CNTHPIRQ[4]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTHVIRQ[4]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTPNSIRQ[4]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTPSIRQ[4]	*Signal* on page 2-85	Master	Timer signals to SOC.
CNTVIRQ[4]	*Signal* on page 2-85	Master	Timer signals to SOC.
acp_s	PVBus	Slave	AXI ACP slave port.
broadcastatomic	*Signal* on page 2-85	Slave	CHI defined pins.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-212 Ports (continued)

Name	Protocol	Type	Description
broadcastcachemaint	*Signal* on page 2-85	Slave	ACE defined pins.
broadcastouter	*Signal* on page 2-85	Slave	ACE defined pins.
broadcastpersist	*Signal* on page 2-85	Slave	CHI defined pins.
cfgend[4]	*Signal* on page 2-85	Slave	This signal if for EE bit initialisation.
cfgsdisable	*Signal* on page 2-85	Slave	This signal disables write access to some secure Interrupt Controller registers.
cfgte[4]	*Signal* on page 2-85	Slave	This signal provides default exception handling state.
clk_in	ClockSignal	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the cluster level components run e.g cluster level timers, caches and pmu.
clusterid	*Value* on page 2-85	Slave	The port reads the value in CPU ID register field, bits[11:8] of the MPIDR.
clusterpmuirq	*Signal* on page 2-85	Master	DynamIQ pmu irq
cntvalueb	*CounterInterface* on page 2-84	Slave	Interface to SoC level counter module.
commirq[4]	*Signal* on page 2-85	Master	Interrupt signal from debug communications channel.
core_clk_in[4]	ClockSignal	Slave	The clock signal connected to the core_clk_in port is used to determine the rate at which each core executes instructions.
cpuporeset[4]	*Signal* on page 2-85	Slave	CPU power on reset. Initializes all the processor logic, including debug logic.
cryptodisable[4]	*Signal* on page 2-85	Slave	Disable cryptography extensions after reset.
cti[4]	*v8EmbeddedCrossTrigger_controlprotocol* on page 2-84	Master	Cross trigger matrix port.
ctidbgirq[4]	*Signal* on page 2-85	Master	Cross Trigger Interface (CTI) interrupt trigger output.
dbgen[4]	*Signal* on page 2-85	Slave	External debug interface.
dbgnopwrdwn[4]	*Signal* on page 2-85	Master	No power-down request.
dbgpwrupreq[4]	*Signal* on page 2-85	Master	Debug power up request.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-212 Ports (continued)

Name	Protocol	Type	Description
dev_debug_s	PVBus	Slave	External debug interface.
event	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE.
fiq[4]	*Signal* on page 2-85	Slave	This signal drives the CPUs fast-interrupt handling.
gicv3_redistributor_s[4]	*GICv3Comms* on page 2-84	Slave	GICv3 AXI-stream port.
irq[4]	*Signal* on page 2-85	Slave	This signal drives the CPUs interrupt handling.
memorymapped_debug_s	PVBus	Slave	External debug interface.
niden[4]	*Signal* on page 2-85	Slave	External debug interface.
pchannel_cluster	*PChannel* on page 2-80	Slave	PChannel for cluster.
pchannel_core[4]	*PChannel* on page 2-80	Slave	PChannels for cores
pmuirq[4]	*Signal* on page 2-85	Master	Interrupt signal from performance monitoring unit.
presetdbg	*Signal* on page 2-85	Slave	Initialize the shared debug APB, Cross Trigger Interface (CTI), and Cross Trigger Matrix (CTM) logic.
pvbus_m0	PVBus	Master	The core will generate bus requests on this port.
pvbus_periph_m	PVBus	Master	The core can generate peripheral bus request on this port.
rei[4]	*Signal* on page 2-85	Slave	Per core RAM Error Interrupt.
reset[4]	*Signal* on page 2-85	Slave	Raising this signal will put the core into reset mode.
rvbaraddr[4]	*Value_64* on page 2-85	Slave	Reset vector base address.
sci_m	SystemCoherencyInterface	Master	System coherency interface port, which is used to take the whole cluster into/out-of coherency domain
sei[4]	*Signal* on page 2-85	Slave	Per core System Error physical pins.
spiden[4]	*Signal* on page 2-85	Slave	External debug interface.
spniden[4]	*Signal* on page 2-85	Slave	External debug interface.
sporeset	*Signal* on page 2-85	Slave	A single cluster-wide power on reset signal for all resettable registers in DynamIQ.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-212 Ports (continued)

Name	Protocol	Type	Description
ticks[4]	*InstructionCount* on page 2-84	Master	This port should be connected to one of the two ticks ports on a 'visualisation' component, in order to display a running instruction count.
vcpumntirq[4]	*Signal* on page 2-85	Master	Interrupt signal for virtual CPU maintenance IRQ.
vfiq[4]	*Signal* on page 2-85	Slave	Virtualised FIQ.
vinithi[4]	*Signal* on page 2-85	Slave	This signal controls of the location of the exception vectors at reset.
virq[4]	*Signal* on page 2-85	Slave	Virtualised IRQ.
virtio_s	PVBus	Slave	The virtio coherent port, hooks directly into the L2 system and becomes coherent (assuming attributes are set correctly).
vsei[4]	*Signal* on page 2-85	Slave	Per core virtual System Error physical pins.

Table 3-213 Parameters for ARM_Neoverse-N1

Name	Type	Default value	Description
cpu0.CFGEND	bool	0x0	Endianness configuration at reset. 0, little endian. 1, big endian.
cpu0.CFGTE	bool	0x0	Instruction set state when resetting into AArch32. 0, A32. 1, T32.
cpu0.CRYPTODISABLE	bool	0x0	Disable cryptographic features.
cpu0.RVBARADDR	int	0x0	Value of RVBAR_ELx register.
cpu0.VINITHI	bool	0x0	Reset value of SCTLR.V.
cpu0.enable_trace_special_hlt_imm16	bool	0x0	Enable usage of parameter trace_special_hlt_imm16
cpu0.l2cache-hit_latency	int	0x0	L2 Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
cpu0.l2cache-maintenance_latency	int	0x0	L2 Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
cpu0.l2cache-miss_latency	int	0x0	L2 Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-213 Parameters for ARM_Neoverse-N1 (continued)

Name	Type	Default value	Description
cpu0.l2cache-read_access_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
cpu0.l2cache-read_latency	int	0x0	L2 Cache timing annotation latency for read accesses given in ticks per byte accessed.l2cache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
cpu0.l2cache-size	int	0x80000	L2 Cache size in bytes.
cpu0.l2cache-snoop_data_transfer_latency	int	0x0	L2 Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
cpu0.l2cache-snoop_issue_latency	int	0x0	L2 Cache timing annotation latency for snoop accesses issued by this cache in total ticks. This is only used when dcache-state_modelled=true.
cpu0.l2cache-write_access_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per access. If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l2cache-write_latency is set. This is only used when dcache-state_modelled=true.
cpu0.l2cache-write_latency	int	0x0	L2 Cache timing annotation latency for write accesses given in ticks per byte accessed. l2cache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
cpu0.max_code_cache_mb	int	0x100	Maximum size of the simulation code cache (MiB). For platforms with more than 2 cores this limit will be scaled down. (e.g 1/8 for 16 or more cores)
cpu0.min_sync_level	int	0x0	Force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)
cpu0.semihosting-A32_HLT	int	0xf000	A32 HLT number for semihosting calls.
cpu0.semihosting-A64_HLT	int	0xf000	A64 HLT number for semihosting calls.
cpu0.semihosting-ARM_SVC	int	0x123456	A32 SVC number for semihosting calls.
cpu0.semihosting-T32_HLT	int	0x3c	T32 HLT number for semihosting calls.
cpu0.semihosting-Thumb_SVC	int	0xab	T32 SVC number for semihosting calls.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-213 Parameters for ARM_Neoverse-N1 (continued)

Name	Type	Default value	Description
cpu0.semihosting-cmd_line	string	""	Command line available to semihosting calls.
cpu0.semihosting-cwd	string	""	Base directory for semihosting file access.
cpu0.semihosting-enable	bool	0x1	Enable semihosting SVC/HLT traps.
cpu0.semihosting-heap_base	int	0x0	Virtual address of heap base.
cpu0.semihosting-heap_limit	int	0xf000000	Virtual address of top of heap.
cpu0.semihosting-stack_base	int	0x10000000	Virtual address of base of descending stack.
cpu0.semihosting-stack_limit	int	0xf000000	Virtual address of stack limit.
cpu0.trace_special_hlt_imm16	int	0xf000	For this HLT number, IF enable_trace_special_hlt_imm16=true, skip performing usual HLT execution but call MTI trace if registered
cpu0.vfp-enable_at_reset	bool	0x0	Enable VFP in CPACR, CPPWR, NSACR at reset. Warning: Arm recommends going through the implementation's suggested VFP power-up sequence!

Table 3-214 Parameters for Cluster_ARM_Neoverse-N1

Name	Type	Default value	Description
BROADCASTATOMIC	bool	0x1	Enable broadcasting of atomic operation. The broadcastatomic signal will override this value if used
BROADCASTCACHEMAINT	bool	0x1	Enable broadcasting of cache maintenance operations to downstream caches. The broadcastcachemaint signal will override this value if used.
BROADCASTOUTER	bool	0x1	Enable broadcasting of Outer Shareable transactions. The broadcastouter signal will override this value if used.
BROADCASTPERSIST	bool	0x1	Enable broadcasting of cache clean to the point of persistence operations. The broadcastpersist signal will override this value if used
CLUSTER_ID	int	0x0	Processor cluster ID value
GICDISABLE	bool	0x1	Disable the new style GICv3 CPU interface in each core model. Should be left enabled unless the platform contains a GICv3 distributor.
cpi_div	int	0x1	Divider for calculating CPI (Cycles Per Instruction)
cpi_mul	int	0x1	Multiplier for calculating CPI (Cycles Per Instruction)
dcache-hit_latency	int	0x0	L1 D-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-214 Parameters for Cluster_ARM_Neoverse-N1 (continued)

Name	Type	Default value	Description
dcache-maintenance_latency	int	0x0	L1 D-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
dcache-miss_latency	int	0x0	L1 D-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
dcache-prefetch_enabled	bool	0x0	Enable simulation of data cache prefetching. This is only used when dcache-state_modelled=true
dcache-read_access_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per access (of size dcache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
dcache-read_latency	int	0x0	L1 D-Cache timing annotation latency for read accesses given in ticks per byte accessed.dcache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
dcache-snoop_data_transfer_latency	int	0x0	L1 D-Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
dcache-state_modelled	bool	0x0	Set whether D-cache has stateful implementation
dcache-write_access_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per access (of size dcache-write_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if dcache-write_latency is set. This is only used when dcache-state_modelled=true.
dcache-write_latency	int	0x0	L1 D-Cache timing annotation latency for write accesses given in ticks per byte accessed. dcache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
default_opmode	int	0x4	Operating mode of DynamIQ coming out of reset. 0: SFONLY ON, 1: 1/4 CACHE ON, 2: 1/2 CACHE ON, 3: 3/4 CACHE ON, 4: FULL CACHE ON

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-214 Parameters for Cluster_ARM_Neoverse-N1 (continued)

Name	Type	Default value	Description
diagnostics	bool	0x0	Enable DynamIQ diagnostic messages
enable_simulation_performance_optimizations	bool	0x1	With this option enabled, the model will run more quickly, but be less accurate to exact CPU behavior. The model will still be functionally accurate for software, but may increase differences seen between hardware behavior and model behavior for certain workloads (it changes the micro-architectural value of stage12_tlb_size parameter to 1024).
ext_abort_device_read_is_sync	bool	0x0	Synchronous reporting of device-nGnRE read external aborts
ext_abort_device_write_is_sync	bool	0x0	Synchronous reporting of device-nGnRE write external aborts
ext_abort_so_read_is_sync	bool	0x0	Synchronous reporting of device-nGnRnE read external aborts
ext_abort_so_write_is_sync	bool	0x0	Synchronous reporting of device-nGnRnE write external aborts
has_coherent_icache	bool	0x0	Whether icache invalidation to the point of unification is required for instruction to data coherence. true - Invalidate operations not required
has_peripheral_port	bool	0x0	If true, additional AXI peripheral port is configured.
has_statistical_profiling	bool	0x1	Whether Statistical Based Profiling is implemented
icache-hit_latency	int	0x0	L1 I-Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when icache-state_modelled=true.
icache-maintenance_latency	int	0x0	L1 I-Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when icache-state_modelled=true.
icache-miss_latency	int	0x0	L1 I-Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when icache-state_modelled=true.
icache-prefetch_enabled	bool	0x0	Enable simulation of instruction cache prefetching. This is only used when icache-state_modelled=true.
icache-read_access_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per access (of size icache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if icache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when icache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-214 Parameters for Cluster_ARM_Neoverse-N1 (continued)

Name	Type	Default value	Description
icache-read_latency	int	0x0	L1 I-Cache timing annotation latency for read accesses given in ticks per byte accessed.icache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when icache-state_modelled=true.
icache-state_modelled	bool	0x0	Set whether I-cache has stateful implementation
l3cache-hit_latency	int	0x0	L3 Cache timing annotation latency for hit. Intended to model the tag-lookup time. This is only used when dcache-state_modelled=true.
l3cache-maintenance_latency	int	0x0	L3 Cache timing annotation latency for cache maintenance operations given in total ticks. This is only used when dcache-state_modelled=true.
l3cache-miss_latency	int	0x0	L3 Cache timing annotation latency for miss. Intended to model the time for failed tag-lookup and allocation of intermediate buffers. This is only used when dcache-state_modelled=true.
l3cache-read_access_latency	int	0x0	L3 Cache timing annotation latency for read accesses given in ticks per access (of size l3cache-read_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l3cache-read_latency is set. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus, this is only used when dcache-state_modelled=true.
l3cache-read_latency	int	0x0	L3 Cache timing annotation latency for read accesses given in ticks per byte accessed.l3cache-read_access_latency must be set to 0 for per-byte latencies to be applied. This is in addition to the hit or miss latency, and intended to correspond to the time taken to transfer across the cache upstream bus. This is only used when dcache-state_modelled=true.
l3cache-size	int	0x100000	L3 Cache size in bytes.
l3cache-snoop_data_transfer_latency	int	0x0	L3 Cache timing annotation latency for received snoop accesses that perform a data transfer given in ticks per byte accessed. This is only used when dcache-state_modelled=true.
l3cache-snoop_issue_latency	int	0x0	L3 Cache timing annotation latency for snoop accesses issued by this cache in total ticks. This is only used when dcache-state_modelled=true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-214 Parameters for Cluster_ARM_Neoverse-N1 (continued)

Name	Type	Default value	Description
l3cache-write_access_latency	int	0x0	L3 Cache timing annotation latency for write accesses given in ticks per access (of size l3cache-write_bus_width_in_bytes). If this parameter is non-zero, per-access latencies will be used instead of per-byte even if l3cache-write_latency is set. This is only used when dcache-state_modelled=true.
l3cache-write_latency	int	0x0	L3 Cache timing annotation latency for write accesses given in ticks per byte accessed. l3cache-write_access_latency must be set to 0 for per-byte latencies to be applied. This is only used when dcache-state_modelled=true.
pchannel_treat_simreset_as_poreset	bool	0x0	Register core as ON state to cluster with simulation reset.
periph_address_end	int	0x0	End address for peripheral port address range exclusive(corresponds to AENDMP input signal).
periph_address_start	int	0x0	Start address for peripheral port address range inclusive(corresponds to ASTARTMP input signal).
ptw_latency	int	0x0	Page table walker latency for TA (Timing Annotation), expressed in simulation ticks
tlb_latency	int	0x0	TLB latency for TA (Timing Annotation), expressed in simulation ticks
treat-dcache-cmos-to-pou-as-nop	bool	0x0	Whether dcache invalidation to the point of unification is required for instruction to data coherence. true - Invalidate operations not required
walk_cache_latency	int	0x0	Walk cache latency for TA (Timing Annotation), expressed in simulation ticks

3.4.48 ARMSC000CT

ARMSC000CT CPU component. This model is written in C++ and models version r0p1 of the RTL.

ARMSC000CT contains the following CADI targets:

- ARM_SC000

ARMSC000CT contains the following MTI components:

- *4.46 ARM_SC000 - trace* on page 4-2350
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402
- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758

Differences between the model and the RTL

The model has the following limitations:

- It does not implement any security features.
- Only bit[0] of the Auxiliary Control Register is supported for read/write. No functionality is implemented.
- The Security Features Control Register read/write access is supported using SECKEY. No functionality is implemented.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-215 Ports

Name	Protocol	Type	Description
ahbd	PVBus	Slave	Debug AHB - core bus slave driven by the DAP.
bigend	*Signal* on page 2-85	Slave	Configure big endian data format.
clk_in	ClockSignal	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the core executes instructions.
currpri	*Value* on page 2-85	Master	Current execution priority.
edbgrq	*Signal* on page 2-85	Slave	External debug request.
event	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE and corresponds to the RTL TXEV and RXEV signals.
intisr[32]	*Signal* on page 2-85	Slave	This signal array delivers signals to the NVIC.
intnmi	*Signal* on page 2-85	Slave	Configure non maskable interrupt.
io_port_in	PVBus	Slave	I/O port pair. See the documentation for the io_port_out port.
io_port_out	PVBus	Master	I/O port pair. Used if IOP is true. Transactions from io_port_out which do not "match" should be returned via io_port_in. For performance reasons, the I/O port interface is not modelled directly. Instead, a simple PVBus gasket is inserted at the point in the memory system where the I/O port would be. In hardware, a device would be attached to the port which would tell the CPU whether it would like to intercept each transaction, given its address. This can be modelled in a performant manner by connecting a PVBusMapper-based device to io_port_out which intercepts transactions of interest and passes other transactions back to the CPU via io_port_in. Your I/O port device model is also responsible for aborting transactions which would be aborted on hardware (e.g. exclusives) if necessary.
lockup	*Signal* on page 2-85	Master	Asserted when the processor is in lockup state.
poreset	*Signal* on page 2-85	Slave	Raising this signal will do a power-on reset of the core.
pv_ppbus_m	PVBus	Master	The core will generate External Private Peripheral Bus requests on this port.
pvbus_m	PVBus	Master	The core will generate bus requests on this port.
sleepdeep	*Signal* on page 2-85	Master	Asserted when the processor is in deep sleep.
sleeping	*Signal* on page 2-85	Master	Asserted when the processor is in sleep.
stcalib	*Value* on page 2-85	Slave	This is the calibration value for the SysTick timer.
stclk	ClockSignal	Slave	This is the reference clock for the SysTick timer.
sysreset	*Signal* on page 2-85	Slave	Raising this signal will put the core into reset mode (but does not reset the debug logic).
sysresetreq	*Signal* on page 2-85	Master	Asserted to indicate that a reset is required.
ticks	*InstructionCount* on page 2-84	Master	Port allowing the number of instructions since startup to be read from the CPU.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-216 Parameters for ARM_SC000

Name	Type	Default value	Description
BIGENDINIT	bool	0x0	Initialize processor to big endian mode
BKPT	int	0x4	Number of breakpoint unit comparators implemented
DBG	bool	0x1	Set whether debug extensions are implemented
IOP	bool	0x0	Send all d-side transactions to the port, io_port_out. Transactions which do not match should be returned to the port, io_port_in
IRQDIS	int	0x0	IRQ line disable mask. Bit n of this 32-bit parameter disables IRQ[n]
NUM_IRQ	int	0x20	Number of user interrupts
NUM_MPU_REGION	int	0x0	Number of MPU regions
SYST	bool	0x1	Enable support for SysTick timer functionality
USER	bool	0x1	Enable support for Unprivileged/Privileged Extension
VTOR	bool	0x1	Include Vector Table Offset Register
WIC	bool	0x1	Include support for WIC-mode deep sleep
WPT	int	0x2	Number of watchpoint unit comparators implemented
cpi_div	int	0x1	divider for calculating CPI (Cycles Per Instruction)
cpi_mul	int	0x1	multiplier for calculating CPI (Cycles Per Instruction)
master_id	int	0x0	Master ID presented in bus transactions
min_sync_level	int	0x0	force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)
scheduler_mode	int	0x0	Control the interleaving of instructions in this processor (0=default long quantum, 1=low latency mode, short quantum and signal checking, 2=lock-breaking mode, long quantum with additional context switches near load-exclusive instructions, 3=ISSCompare)
semihosting-Thumb_SVC	int	0xab	T32 SVC number for semihosting
semihosting-cmd_line	string	""	Command line available to semihosting SVC calls
semihosting-cwd	string	""	Base directory for semihosting file access.
semihosting-enable	bool	0x1	Enable semihosting SVC traps. Applications that do not use semihosting must set this parameter to false.
semihosting-heap_base	int	0x0	Virtual address of heap base
semihosting-heap_limit	int	0x10700000	Virtual address of top of heap
semihosting-prefix	bool	0x0	Prefix semihosting output with target instance name
semihosting-stack_base	int	0x10700000	Virtual address of base of descending stack
semihosting-stack_limit	int	0x10800000	Virtual address of stack limit

3.4.49 ARMSC300CT

ARMSC300CT CPU component. This model is written in C++ and models version r0p1 of the RTL.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ARMSC300CT contains the following CADI targets:

- ARM_SC300

ARMSC300CT contains the following MTI components:

- *4.47 ARM_SC300 - trace* on page 4-2381
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402
- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758

Differences between the model and the RTL

The model has the following limitations:

- It does not implement any security features.
- The Trash Register is implemented as RAZ/WI.
- Only bit[0] of the Auxiliary Control Register is supported for read/write. No functionality is implemented.
- The Security Features Control Register read/write access is supported using SECKEY. No functionality is implemented.

Implementation of ITM in M-class models

This model has a parameter that enables partial support for Instrumentation Trace Macrocell (ITM). In hardware or RTL, trace data from ITM is sent in packets to the trace block serially using a single pin or wire. In the model, if it is enabled, the ITM trace data is output using an MTI trace source called `ITM`. The `ITM` trace source has an `ITM_PACKET_TYPE` field. The following table shows which packet types the model supports:

Table 3-217 ITM_PACKET_TYPE field values that the model supports

Field value	Description	Supported by model
`ITM_SYNC`	Synchronization packet	Not supported.
`ITM_P_OVERFLOW`	Protocol: Overflow packet	Not supported.
`ITM_P_LOCAL_TIMESTAMP`	Protocol: Local timestamp packets	Not supported.
`ITM_P_GLOBAL_TIMESTAMP`	Protocol: Global timestamp packets	Not supported.
`ITM_P_EXTEN`	Protocol: Extension packet	Not supported.
`ITM_S_INSTRUMENTATION`	Source: Instrumentation packet	Supported.
`ITM_S_DWT_EVENT_COUNTER`	Hardware source: Event counter wrapping	Not supported.
`ITM_S_DWT_EXCEPTION`	Hardware source: Exception tracing	Supported.
`ITM_S_DWT_PC_SAMPLING`	Hardware source: PC sampling	Not supported.
`ITM_S_DWT_DATA_PC_TRACE`	Hardware source: DWT Data trace PC value	Supported.
`ITM_S_DWT_DATA_ADDRESS_TRACE`	Hardware source: DWT Data trace address value	Supported.
`ITM_S_DWT_DATA_DATA_TRACE`	Hardware source: DWT Data trace DATA value	Supported.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-218 Ports

Name	Protocol	Type	Description
ahbd	PVBus	Slave	Debug AHB - core bus slave driven by the DAP.
auxfault	*Value* on page 2-85	Slave	This is wired to the Auxiliary Fault Status Register.
bigend	*Signal* on page 2-85	Slave	Configure big endian data format.
clk_in	ClockSignal	Slave	The clock signal connected to the clk_in port is used to determine the rate at which the core executes instructions.
currpri	*Value* on page 2-85	Master	Current execution priority.
edbgrq	*Signal* on page 2-85	Slave	External debug request.
event	*Signal* on page 2-85	Peer	This peer port of event input (and output) is for wakeup from WFE and corresponds to the RTL TXEV and RXEV signals.
intisr[240]	*Signal* on page 2-85	Slave	This signal array delivers signals to the NVIC.
intnmi	*Signal* on page 2-85	Slave	Configure non maskable interrupt.
lockup	*Signal* on page 2-85	Master	Asserted when the processor is in lockup state.
poreset	*Signal* on page 2-85	Slave	Raising this signal will do a power-on reset of the core.
pv_ppbus_m	PVBus	Master	The core will generate External Private Peripheral Bus requests on this port.
pvbus_m	PVBus	Master	The core will generate bus requests on this port.
sleepdeep	*Signal* on page 2-85	Master	Asserted when the processor is in deep sleep.
sleeping	*Signal* on page 2-85	Master	Asserted when the processor is in sleep.
stcalib	*Value* on page 2-85	Slave	This is the calibration value for the SysTick timer.
stclk	ClockSignal	Slave	This is the reference clock for the SysTick timer.
sysreset	*Signal* on page 2-85	Slave	Raising this signal will put the core into reset mode (but does not reset the debug logic).
sysresetreq	*Signal* on page 2-85	Master	Asserted to indicate that a reset is required.
ticks	*InstructionCount* on page 2-84	Master	Port allowing the number of instructions since startup to be read from the CPU.

Table 3-219 Parameters for ARM_SC300

Name	Type	Default value	Description
BB_PRESENT	bool	0x1	Enable bitbanding
BIGENDINIT	bool	0x0	Initialize processor to big endian mode
DBGLVL	int	0x3	0: No debug; 1: Minimal debug; 2: Full debug without DWT address matching; 3: Full debug support with DWT data-comparators
LVL_WIDTH	int	0x3	Number of bits of interrupt priority
NUM_IRQ	int	0x10	Number of user interrupts
NUM_MPU_REGION	int	0x8	Number of MPU regions
TRACE_LVL	bool	0x1	Level of instrumentation trace supported. false : No ITM trace included, true: ITM trace included

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-219 Parameters for ARM_SC300 (continued)

Name	Type	Default value	Description
WIC	bool	0x1	Include support for WIC-mode deep sleep
cpi_div	int	0x1	divider for calculating CPI (Cycles Per Instruction)
cpi_mul	int	0x1	multiplier for calculating CPI (Cycles Per Instruction)
master_id	int	0x0	Master ID presented in bus transactions
min_sync_level	int	0x0	force minimum syncLevel (0=off=default,1=syncState,2=postInsnIO,3=postInsnAll)
scheduler_mode	int	0x0	Control the interleaving of instructions in this processor (0=default long quantum, 1=low latency mode, short quantum and signal checking, 2=lock-breaking mode, long quantum with additional context switches near load-exclusive instructions, 3=ISSCompare)
semihosting-Thumb_SVC	int	0xab	T32 SVC number for semihosting
semihosting-cmd_line	string	""	Command line available to semihosting SVC calls
semihosting-cwd	string	""	Base directory for semihosting file access.
semihosting-enable	bool	0x1	Enable semihosting SVC traps. Applications that do not use semihosting must set this parameter to false.
semihosting-heap_base	int	0x0	Virtual address of heap base
semihosting-heap_limit	int	0x10700000	Virtual address of top of heap
semihosting-prefix	bool	0x0	Prefix semihosting output with target instance name
semihosting-stack_base	int	0x10700000	Virtual address of base of descending stack
semihosting-stack_limit	int	0x10800000	Virtual address of stack limit

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

3.5 Media components

This section describes the Media components.

This section contains the following subsections:

- *3.5.1 D71* on page 3-601.
- *3.5.2 DP500* on page 3-602.
- *3.5.3 DP500x2* on page 3-603.
- *3.5.4 DP550* on page 3-604.
- *3.5.5 DP550x2* on page 3-605.
- *3.5.6 DP650* on page 3-606.
- *3.5.7 DP650x2* on page 3-607.
- *3.5.8 Mali_G51* on page 3-608.
- *3.5.9 Mali_G71* on page 3-609.
- *3.5.10 Mali_G72* on page 3-610.
- *3.5.11 Mali_G76* on page 3-612.
- *3.5.12 V61* on page 3-613.
- *3.5.13 V550* on page 3-614.

3.5.1 D71

ARM D71 Display Processor. This model is written in C++.

D71 contains the following CADI targets:

- D71

D71 contains the following MTI components:

- *4.61 D71 - trace* on page 4-2439
- *4.107 PVBusSlave - trace* on page 4-2760

About D71

The model has the following limitations:

- No support for trusted layers.
- No support for image enhancements.
- No coprocessor support for HDR processing.
- No QoS support.
- The following configuration parameters are not available:
  - `CONFIG_MAX_LINE_SIZE`
  - `CONFIG_DISPLAY_TBU_EN`. TBUs are integrated separately using the given ports.
  - `CONFIG_AFBC_DMA_EN`. The ADU is present. If it is not used, do not program it.

Table 3-220 Ports

Name	Protocol	Type	Description
`apb_pvbus_s_adu`	`PVBus`	Slave	Slave port for register access.
`apb_pvbus_s_dpu`	`PVBus`	Slave	-
`axi_pvbus_adu_m`	`PVBus`	Master	Master AXI port for the AFBC unit
`axi_pvbus_lpu_m[2]`	`PVBus`	Master	Master AXI ports for pipelines
`display[2]`	*`LCD` on page 2-72*	Master	LCD ports for display outputs
`irq0_gcu_out`	*`Signal` on page 2-85*	Master	Shared interrupt owned by the GCU
`irq1_adu_out`	*`Signal` on page 2-85*	Master	Interrupt signal for the ADU block
`pixelclock_in[2]`	`ClockSignal`	Slave	Pixel clock inputs for the display outputs

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-220 Ports (continued)

Name	Protocol	Type	Description
pvbus_tbu_m[2]	PVBus	Master	Master ports for connection to TBU (SMMUv3)
pvbus_tbu_s[2]	PVBus	Slave	Slave ports for loopback from TBU (SMMUv3)
reset_signal	*Signal* on page 2-85	Slave	Reset signal.

3.5.2 DP500

ARM DP500 Display Processor. This model is written in C++.

DP500 contains the following CADI targets:

- DP500

DP500 contains the following MTI components:
- *4.65 DP500 - trace* on page 4-2455
- *4.106 PVBusMaster - trace* on page 4-2759
- *4.107 PVBusSlave - trace* on page 4-2760

About DP500

This model has basic support for the display and scaling engines. Connect it to a visualization component to view LCD output. This is the single display configuration of DP500. For the dual display configuration, use DP500x2.

It passes tests as part of a booting Android kernel and running under the control of the official DP500 drivers.

It provides the following functionality:

- All RGB and YUV format parsing.
- Color adjustment in *Display Engine* (DE).
- Nearest neighbor scaling.
- All layers.
- Alpha blending.
- Memory writeback.
- Inverse gamma adjustment.
- Basic layer (overlay) and register security semantics.

The model has the following limitations:
- No support for polyphase scaling algorithm. Falls back to nearest neighbor when configured to do so.
- No support for 3D or interlaced video.
- No support for image enhancing functionality.
- No colorspace conversion support.
- No support for two plane YUV memory writeback.

Table 3-221 Ports

Name	Protocol	Type	Description
clk_in	ClockSignal	Slave	Master clock input, typically 24MHz, to drive pixel clock timing.
display	*LCD* on page 2-72	Master	Connection to visualization component.
intr	*Signal* on page 2-85	Master	Interrupt signal.
intr_se	*Signal* on page 2-85	Master	Interrupt signal from scaling engine.
pvbus_m	PVBus	Master	Bus for processor 0.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-221 Ports (continued)

Name	Protocol	Type	Description
pvbus_s	PVBus	Slave	Slave port for register access.
reset_signal	*Signal* on page 2-85	Slave	Reset signal.

3.5.3 DP500x2

ARM DP500 Display Processor x2. This model is written in LISA+.

DP500x2 contains the following CADI targets:

- DP500
- DP500x2

DP500x2 contains the following MTI components:

- *4.65 DP500 - trace* on page 4-2455
- *4.106 PVBusMaster - trace* on page 4-2759
- *4.107 PVBusSlave - trace* on page 4-2760

About DP500x2

This component is a model of the dual display configuration of the DP500 Display Processor, with basic support for the display and scaling engines. Connect it to a visualization component to view LCD output.

The model provides the following functionality:

- All RGB and YUV format parsing.
- Color adjustment in *Display Engine* (DE).
- Nearest neighbor scaling.
- All layers.
- Alpha blending.
- Memory writeback.
- Inverse gamma adjustment.
- Basic layer (overlay) and register security semantics.

The model has the following limitations:

- No support for polyphase scaling algorithm. Falls back to nearest neighbor when configured to do so.
- No support for 3D or interlaced video.
- No support for image enhancing functionality.
- No colorspace conversion support.
- No support for two plane YUV memory writeback.

Table 3-222 Ports

Name	Protocol	Type	Description
dc_de_interrupt[2]	*Signal* on page 2-85	Master	Interrup signalling from display engines.
dc_pvbus_m[2]	PVBus	Master	Bus for processor 0 and 1.
dc_se_interrupt[2]	*Signal* on page 2-85	Master	Interrupt signalling from scaling engines.
display[2]	*LCD* on page 2-72	Master	Connection to visualization component.
dp0_clk_in	ClockSignal	Slave	Clock signal for DP0.
dp1_clk_in	ClockSignal	Slave	Clock signal for DP1.
pvbus_s	PVBus	Slave	Slave port for register access.
reset_signal	*Signal* on page 2-85	Slave	Reset signal.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-223 Parameters for DP500x2

Name	Type	Default value	Description
disable_snooping_dma	bool	0x0	Disable DMA snooping
force_frame_rate	int	0x0	If 0 - the input clock is used as PXLCLOCK, if >0 then the model ensures the screen display is refreshed n times per simulated second

3.5.4 DP550

ARM DP550 Display Processor. This model is written in C++.

DP550 contains the following CADI targets:

- DP550

DP550 contains the following MTI components:

- *4.66 DP550 - trace* on page 4-2457
- *4.106 PVBusMaster - trace* on page 4-2759
- *4.107 PVBusSlave - trace* on page 4-2760

About DP550

This component is a model of the DP550 Display Processor. Connect it to a visualization component to view LCD output. This is the single display configuration of DP550. For the dual display configuration, use DP550x2.

The model provides the following functionality:

- All RGB and YUV format parsing.
- Color adjustment in *Display Engine* (DE).
- Nearest neighbor scaling.
- All layers.
- Alpha blending.
- Memory writeback.
- Inverse gamma adjustment.
- Basic layer (overlay) and register security semantics.

The model has the following limitations:

- No support for polyphase scaling algorithm. Falls back to nearest neighbor when configured to do so.
- No support for 3D or interlaced video.
- No support for image enhancing functionality.
- No colorspace conversion support.
- No support for two plane YUV memory writeback.

Table 3-224 Ports

Name	Protocol	Type	Description
clk_in	ClockSignal	Slave	Master clock input, typically 24MHz, to drive pixel clock timing.
display	*LCD on page 2-72*	Master	Connection to visualization component.
intr	*Signal on page 2-85*	Master	Interrupt signal.
intr_se	*Signal on page 2-85*	Master	Interrupt signal from scaling engine.
pvbus_m	PVBus	Master	Bus for processor 0.
pvbus_s	PVBus	Slave	Slave port for register access.
reset_signal	*Signal on page 2-85*	Slave	Reset signal.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

3.5.5 DP550x2

ARM DP550 Display Processor x2. This model is written in LISA+.

DP550x2 contains the following CADI targets:

- DP550
- DP550x2

DP550x2 contains the following MTI components:

- *4.66 DP550 - trace* on page 4-2457
- *4.106 PVBusMaster - trace* on page 4-2759
- *4.107 PVBusSlave - trace* on page 4-2760

About DP550x2

This component is a model of the dual display configuration of the DP550 Display Processor. Connect it to a visualization component to view LCD output.

The model provides the following functionality:

- All RGB and YUV format parsing.
- Color adjustment in *Display Engine* (DE).
- Nearest neighbor scaling.
- All layers.
- Alpha blending.
- Memory writeback.
- Inverse gamma adjustment.
- Basic layer (overlay) and register security semantics.

The model has the following limitations:

- No support for polyphase scaling algorithm. Falls back to nearest neighbor when configured to do so.
- No support for 3D or interlaced video.
- No support for image enhancing functionality.
- No colorspace conversion support.
- No support for two plane YUV memory writeback.

Table 3-225 Ports

Name	Protocol	Type	Description
dc_de_interrupt[2]	*Signal* on page 2-85	Master	Interrup signalling from display engines.
dc_pvbus_m[2]	PVBus	Master	Bus for processor 0 and 1.
dc_se_interrupt[2]	*Signal* on page 2-85	Master	Interrupt signalling from scaling engines.
display[2]	*LCD* on page 2-72	Master	Connection to visualization component.
dp0_clk_in	ClockSignal	Slave	Clock signal for DP0.
dp1_clk_in	ClockSignal	Slave	Clock signal for DP1.
pvbus_s	PVBus	Slave	Slave port for register access.
reset_signal	*Signal* on page 2-85	Slave	Reset signal.

Table 3-226 Parameters for DP550x2

Name	Type	Default value	Description
disable_snooping_dma	bool	0x0	Disable DMA snooping
force_frame_rate	int	0x0	If 0 - the input clock is used as PXLCLOCK, if >0 then the model ensures the screen display is refreshed n times per simulated second

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-226 Parameters for DP550x2 (continued)

Name	Type	Default value	Description
set_primary_nprot_nsaid	int	0xc00ba98	Configurable NPROT_NSAID for non-secure NSAID values(primary)
set_primary_nprot_streamid	int	0xc00ba98	Configurable NPROT_STREAMID for non-secure STREAMID values(primary)
set_primary_prot_nsaid	int	0x4003210	Configurable PROT_NSAID for secure NSAID values(primary)
set_primary_prot_streamid	int	0x4003210	Configurable PROT_STREAMID for secure STREAMID values(primary)
set_secondary_nprot_nsaid	int	0xc00ba98	Configurable NPROT_NSAID for non-secure NSAID values(secondary)
set_secondary_nprot_streamid	int	0xc00ba98	Configurable NPROT_STREAMID for non-secure STREAMID values(secondary)
set_secondary_prot_nsaid	int	0x4003210	Configurable PROT_NSAID for secure NSAID values(secondary)
set_secondary_prot_streamid	int	0x4003210	Configurable PROT_STREAMID for secure STREAMID values(secondary)

3.5.6 DP650

ARM DP650 Display Processor. This model is written in C++.

DP650 contains the following CADI targets:

- DP650

DP650 contains the following MTI components:

- *4.67 DP650 - trace* on page 4-2460
- *4.106 PVBusMaster - trace* on page 4-2759
- *4.107 PVBusSlave - trace* on page 4-2760

About DP650

This component is a model of the DP650 Display Processor. Connect it to a visualization component to view LCD output. This is the single display configuration of DP650. For the dual display configuration, use DP650x2.

The model has the following limitations:

- No support for the polyphase scaling algorithm, it always uses nearest neighbor.
- No support for 3D or interlaced video.
- No support for image enhancing functionality.
- No YUV 2-plane support for memory writeback.
- No color space conversion support.

Table 3-227 Ports

Name	Protocol	Type	Description
clk_in	ClockSignal	Slave	Master clock input, typically 24MHz, to drive pixel clock timing.
display	*LCD* on page 2-72	Master	Connection to visualization component.
intr	*Signal* on page 2-85	Master	Interrupt signal.
intr_se	*Signal* on page 2-85	Master	Interrupt signal from scaling engine.
pvbus_m	PVBus	Master	Bus for processor 0.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-227 Ports (continued)

Name	Protocol	Type	Description
pvbus_s	PVBus	Slave	Slave port for register access.
reset_signal	*Signal* on page 2-85	Slave	Reset signal.

3.5.7 DP650x2

ARM DP650 Display Processor x2. This model is written in LISA+.

DP650x2 contains the following CADI targets:

- DP650
- DP650x2

DP650x2 contains the following MTI components:

- *4.67 DP650 - trace* on page 4-2460
- *4.106 PVBusMaster - trace* on page 4-2759
- *4.107 PVBusSlave - trace* on page 4-2760

About DP650x2

This component is a model of the dual display configuration of the DP650 Display Processor. To view LCD output, connect it to a visualization component.

The model has the following limitations:

- No support for the polyphase scaling algorithm, it always uses nearest neighbor.
- No support for 3D or interlaced video.
- No support for image enhancing functionality.
- No YUV 2-plane support for memory writeback.
- No color space conversion support.

Table 3-228 Ports

Name	Protocol	Type	Description
dc_de_interrupt[2]	*Signal* on page 2-85	Master	Interrup signalling from display engines.
dc_pvbus_m[2]	PVBus	Master	Bus for processor 0 and 1.
dc_se_interrupt[2]	*Signal* on page 2-85	Master	Interrupt signalling from scaling engines.
display[2]	*LCD* on page 2-72	Master	Connection to visualization component.
dp0_clk_in	ClockSignal	Slave	Clock signal for DP0.
dp1_clk_in	ClockSignal	Slave	Clock signal for DP1.
pvbus_s	PVBus	Slave	Slave port for register access.
reset_signal	*Signal* on page 2-85	Slave	Reset signal.

Table 3-229 Parameters for DP650x2

Name	Type	Default value	Description
disable_snooping_dma	bool	0x0	Disable DMA snooping
force_frame_rate	int	0x0	If 0 - the input clock is used as PXLCLOCK, if >0 then the model ensures the screen display is refreshed n times per simulated second
set_primary_nprot_nsaid	int	0xc00ba98	Configurable NPROT_NSAID for non-secure NSAID values(primary)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-229 Parameters for DP650x2 (continued)

Name	Type	Default value	Description
set_primary_nprot_streamid	int	0xc00ba98	Configurable NPROT_STREAMID for non-secure STREAMID values(primary)
set_primary_prot_nsaid	int	0x4003210	Configurable PROT_NSAID for secure NSAID values(primary)
set_primary_prot_streamid	int	0x4003210	Configurable PROT_STREAMID for secure STREAMID values(primary)
set_secondary_nprot_nsaid	int	0xc00ba98	Configurable NPROT_NSAID for non-secure NSAID values(secondary)
set_secondary_nprot_streamid	int	0xc00ba98	Configurable NPROT_STREAMID for non-secure STREAMID values(secondary)
set_secondary_prot_nsaid	int	0x4003210	Configurable PROT_NSAID for secure NSAID values(secondary)
set_secondary_prot_streamid	int	0x4003210	Configurable PROT_STREAMID for secure STREAMID values(secondary)

3.5.8 Mali_G51

Arm Mali™-G51 GPU. This model is written in C++.

Mali_G51 contains the following CADI targets:

- Mali_G51

Mali_G51 contains the following MTI components:

- *4.87 Mali_G51 - trace* on page 4-2724
- *4.106 PVBusMaster - trace* on page 4-2759
- *4.107 PVBusSlave - trace* on page 4-2760

About Mali_G51

The model has the following limitations:

- It does not support Armv8-style page tables.
- It does not execute GPU shader programs.
- It does not validate all register values or job descriptors.

This model outputs values for all Mali PMU hardware counters. These values are not meaningful. Their purpose is to enable you to use the model early in the development process to check that your system is correctly configured to support performance analysis tools, for instance Streamline.

The counter is a 32-bit bitfield, with the following bit assignments:

[31:28]

Identifies which Mali GPU instance generated this value, typically zero for a system with a single GPU.

[27:24]

Identifies which hardware block is the source of the counter. It can have one of the following values:

0 Job manager.

1 Tiler.

2 L2Cache/Memory system.

3+ Shader core.

[23:16]

The counter number within the block.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

[15:0]

Sawtooth counter that counts from 0 to `0xffff` and then resets to 0. This value ensures that consecutive captures show different values and allows you to check that counter values are changing over time.

Note

- These counter values might change in future.
- The model always generates counter values, even if the GPU is idle.
- Streamline does not display the precise values that the model outputs because it needs to correct them for sampling frequency and other profiling effects. However, their size relative to each other is correct. For example, counters from the job manager are always smaller than those from the memory system.

Table 3-230 Ports

Name	Protocol	Type	Description
gpu_reset	*Signal* on page 2-85	Slave	Resets the GPU at the input Set. The available inputs are Set and Clear.
irq_gpu	*Signal* on page 2-85	Master	The interrupt signal generated from the GPU.
irq_job	*Signal* on page 2-85	Master	The interrupt signal generated from the Job Manager on the GPU.
irq_mmu	*Signal* on page 2-85	Master	The interrupt signal generated from the MMU on the GPU.
prot_mode_m0	*Signal* on page 2-85	Master	Indicates the current state of the GPU. When it outputs: Set: The GPU is in protected mode. Clear: The GPU is in normal mode. Typically, connect this signal to the LabellerForGPUProtMode component. This enables the labeller to add a Non-Secure Access ID to the outgoing transaction on the labeller's pvbus_m port.
pvbus_m	PVBus	Master	The interface for the GPU to access external memory.
pvbus_s	PVBus	Slave	The interface for the CPU to access the GPU registers.

3.5.9 Mali_G71

Arm Mali-G71 GPU. This model is written in C++.

Mali_G71 contains the following CADI targets:

- Mali_G71

Mali_G71 contains the following MTI components:

- *4.88 Mali_G71 - trace* on page 4-2726
- *4.106 PVBusMaster - trace* on page 4-2759
- *4.107 PVBusSlave - trace* on page 4-2760

About Mali_G71

The model has the following limitations:

- It does not support Armv8-style page tables.
- It does not execute GPU shader programs.
- It does not validate all register values or job descriptors.

This model outputs values for all Mali PMU hardware counters. These values are not meaningful. Their purpose is to enable you to use the model early in the development process to check that your system is correctly configured to support performance analysis tools, for instance Streamline.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

The counter is a 32-bit bitfield, with the following bit assignments:

[31:28]
Identifies which Mali GPU instance generated this value, typically zero for a system with a single GPU.

[27:24]
Identifies which hardware block is the source of the counter. It can have one of the following values:

0 Job manager.

1 Tiler.

2 L2Cache/Memory system.

3+ Shader core.

[23:16]
The counter number within the block.

[15:0]

Sawtooth counter that counts from 0 to `0xffff` and then resets to 0. This value ensures that consecutive captures show different values and allows you to check that counter values are changing over time.

Note

- These counter values might change in future.
- The model always generates counter values, even if the GPU is idle.
- Streamline does not display the precise values that the model outputs because it needs to correct them for sampling frequency and other profiling effects. However, their size relative to each other is correct. For example, counters from the job manager are always smaller than those from the memory system.

Table 3-231 Ports

Name	Protocol	Type	Description
gpu_reset	*Signal* on page 2-85	Slave	Resets the GPU at the input Set. The available inputs are Set and Clear.
irq_gpu	*Signal* on page 2-85	Master	The interrupt signal generated from the GPU.
irq_job	*Signal* on page 2-85	Master	The interrupt signal generated from the Job Manager on the GPU.
irq_mmu	*Signal* on page 2-85	Master	The interrupt signal generated from the MMU on the GPU.
prot_mode_m0	*Signal* on page 2-85	Master	Indicates the current state of the GPU. When it outputs: Set: The GPU is in protected mode. Clear: The GPU is in normal mode. Typically, connect this signal to the LabellerForGPUProtMode component. This enables the labeller to add a Non-Secure Access ID to the outgoing transaction on the labeller's pvbus_m port.
pvbus_m	PVBus	Master	The interface for the GPU to access external memory.
pvbus_s	PVBus	Slave	The interface for the CPU to access the GPU registers.

3.5.10 Mali_G72

Arm Mali-G72 GPU. This model is written in C++.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Mali_G72 contains the following CADI targets:

- Mali_G72

Mali_G72 contains the following MTI components:

- *4.89 Mali_G72 - trace* on page 4-2728
- *4.106 PVBusMaster - trace* on page 4-2759
- *4.107 PVBusSlave - trace* on page 4-2760

About Mali_G72

The model has the following limitations:

- It does not support Armv8-style page tables.
- It does not execute GPU shader programs.
- It does not validate all register values or job descriptors.

This model outputs values for all Mali PMU hardware counters. These values are not meaningful. Their purpose is to enable you to use the model early in the development process to check that your system is correctly configured to support performance analysis tools, for instance Streamline.

The counter is a 32-bit bitfield, with the following bit assignments:

[31:28]

Identifies which Mali GPU instance generated this value, typically zero for a system with a single GPU.

[27:24]

Identifies which hardware block is the source of the counter. It can have one of the following values:

0 Job manager.

1 Tiler.

2 L2Cache/Memory system.

3+ Shader core.

[23:16]

The counter number within the block.

[15:0]

Sawtooth counter that counts from 0 to `0xffff` and then resets to 0. This value ensures that consecutive captures show different values and allows you to check that counter values are changing over time.

Note

- These counter values might change in future.
- The model always generates counter values, even if the GPU is idle.
- Streamline does not display the precise values that the model outputs because it needs to correct them for sampling frequency and other profiling effects. However, their size relative to each other is correct. For example, counters from the job manager are always smaller than those from the memory system.

Table 3-232 Ports

Name	Protocol	Type	Description
`gpu_reset`	*Signal* on page 2-85	Slave	Resets the GPU at the input Set. The available inputs are Set and Clear.
`irq_gpu`	*Signal* on page 2-85	Master	The interrupt signal generated from the GPU.
`irq_job`	*Signal* on page 2-85	Master	The interrupt signal generated from the Job Manager on the GPU.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-232 Ports (continued)

Name	Protocol	Type	Description
irq_mmu	*Signal* on page 2-85	Master	The interrupt signal generated from the MMU on the GPU.
prot_mode_m0	*Signal* on page 2-85	Master	Indicates the current state of the GPU. When it outputs: Set: The GPU is in protected mode. Clear: The GPU is in normal mode. Typically, connect this signal to the LabellerForGPUProtMode component. This enables the labeller to add a Non-Secure Access ID to the outgoing transaction on the labeller's pvbus_m port.
pvbus_m	PVBus	Master	The interface for the GPU to access external memory.
pvbus_s	PVBus	Slave	The interface for the CPU to access the GPU registers.

3.5.11 Mali_G76

Arm Mali-G76 GPU. This model is written in C++.

Mali_G76 contains the following CADI targets:

- Mali_G76

Mali_G76 contains the following MTI components:

- *4.90 Mali_G76 - trace* on page 4-2730
- *4.106 PVBusMaster - trace* on page 4-2759
- *4.107 PVBusSlave - trace* on page 4-2760

About Mali_G76

The model has the following limitations:

- It does not support Armv8-style page tables.
- It does not execute GPU shader programs.
- It does not validate all register values or job descriptors.

This model outputs values for all Mali PMU hardware counters. These values are not meaningful. Their purpose is to enable you to use the model early in the development process to check that your system is correctly configured to support performance analysis tools, for instance Streamline.

The counter is a 32-bit bitfield, with the following bit assignments:

[31:28]

Identifies which Mali GPU instance generated this value, typically zero for a system with a single GPU.

[27:24]

Identifies which hardware block is the source of the counter. It can have one of the following values:

0 Job manager.

1 Tiler.

2 L2Cache/Memory system.

3+ Shader core.

[23:16]

The counter number within the block.

[15:0]

Sawtooth counter that counts from 0 to `0xffff` and then resets to 0. This value ensures that consecutive captures show different values and allows you to check that counter values are changing over time.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Note

- These counter values might change in future versions.
- The model always generates counter values, even if the GPU is idle.
- Streamline does not display the precise values that the model outputs because it needs to correct them for sampling frequency and other profiling effects. However, their size relative to each other is correct. For example, counters from the job manager are always smaller than those from the memory system.

Table 3-233 Ports

Name	Protocol	Type	Description
gpu_reset	*Signal* on page 2-85	Slave	Resets the GPU at the input Set. The available inputs are Set and Clear.
irq_gpu	*Signal* on page 2-85	Master	The interrupt signal generated from the GPU.
irq_job	*Signal* on page 2-85	Master	The interrupt signal generated from the Job Manager on the GPU.
irq_mmu	*Signal* on page 2-85	Master	The interrupt signal generated from the MMU on the GPU.
prot_mode_m0	*Signal* on page 2-85	Master	Indicates the current state of the GPU. When it outputs: Set: The GPU is in protected mode. Clear: The GPU is in normal mode. Typically, connect this signal to the LabellerForGPUProtMode component. This enables the labeller to add a Non-Secure Access ID to the outgoing transaction on the labeller's pvbus_m port.
pvbus_m	PVBus	Master	The interface for the GPU to access external memory.
pvbus_s	PVBus	Slave	The interface for the CPU to access the GPU registers.

3.5.12 V61

ARM(r) Mali(tm)-V61 Video Processor. This model is written in C++.

V61 contains the following CADI targets:

- V61

V61 contains the following MTI components:

- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.116 V61 - trace* on page 4-2852

About V61

The model requires an external OpenMAX (OMX) IL implementation for codec functionality. By default, V61 looks for `ffomaxil.dll` on Windows or `libffomaxil.so` on Linux in the model binary's directory or in the Fast Models installation. The default library path can be overridden using the `omx-library-path` parameter.

FFomaxIL is an OMX IL implementation provided by Arm in the TPIP package for convenience. Refer to the TPIP package for more details on FFomaxIL.

When querying the OMX core, V61 searches for the following roles in the list of OpenMAX components:

H.264 decode "video_decoder.avc"
JPEG decode "video_decoder.mjpeg"
MPEG2 decode "video_decoder.mpeg2"
MPEG4 decode "video_decoder.mpeg4"

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VC1 decode	"video_decoder.vc1"
VP8 decode	"video_decoder.vp8"
VP8 encode	"video_encoder.vp8"

Note

To build example platforms containing V61, you must either install the TPIP package, or remove the dependency on FFmpeg and libvpx from the platform's sgproj file, by removing the line containing `V5xx.sgrepo`.

The model has the following limitations:

- No support for HEVC, VP9 and RealVideo decoders.
- No support for 10-bit video output.
- No support for RGB or AFBC input for encoding.
- No profiling support.
- No QoS support.
- Power/Test modes are modeled only as register state changes.

Table 3-234 Ports

Name	Protocol	Type	Description
`apb_s`	`PVBus`	Slave	APB Slave port for register access.
`axi_m`	`PVBus`	Master	AXI master bus for memory accesses.
`clk`	`ClockSignal`	Slave	Master clock, typically 300MHz.
`irq`	*Signal* on page 2-85	Master	IRQ signal to host CPU.
`reset`	*Signal* on page 2-85	Slave	Reset signal.

3.5.13 V550

ARM(r) Mali(tm)-V550 Video Processor. This model is written in C++.

V550 contains the following CADI targets:

- V550

V550 contains the following MTI components:

- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.115 V550 - trace* on page 4-2838

About V550

The model requires an external OpenMAX (OMX) IL implementation for codec functionality. By default, V550 looks for `ffomaxil.dll` on Windows or `libffomaxil.so` on Linux in the model binary's directory or in the Fast Models installation. The default library path can be overridden using the parameter `omx-library-path`.

FFomaxIL is an OMX IL implementation provided by Arm in the Third-Party IP (*TPIP*) package for convenience. Refer to the TPIP package for more details on FFomaxIL.

When querying the OMX core, V550 searches for the following roles in the list of OpenMAX components:

H.264 decode	"video_decoder.avc"
JPEG decode	"video_decoder.mjpeg"
MPEG2 decode	"video_decoder.mpeg2"

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MPEG4 decode	"video_decoder.mpeg4"
VC1 decode	"video_decoder.vc1"
VP8 decode	"video_decoder.vp8"
VP8 encode	"video_encoder.vp8"

Note

To build example platforms containing V550, you must either install the TPIP package, or remove the dependency on FFmpeg and libvpx from the platform's sgproj file, by removing the line containing `V5xx.sgrepo`.

The model has the following limitations:

- No support for HEVC and RealVideo decoders.
- No support for 10-bit video output.
- No profiling support.
- No QoS support.
- Power and Test modes are modeled only as register state changes.

Table 3-235 Ports

Name	Protocol	Type	Description
apb_s	PVBus	Slave	APB Slave port for register access.
axi_m	PVBus	Master	AXI master bus for memory accesses.
clk	ClockSignal	Slave	Master clock, typically 300MHz.
irq	*Signal* on page 2-85	Master	IRQ signal to host CPU.
reset	*Signal* on page 2-85	Slave	Reset signal.

Related information

Fixed Virtual Platforms

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

3.6 Other components

This section describes the Other components.

This section contains the following subsections:

- *3.6.1 AsyncSignal* on page 3-616.
- *3.6.2 GICv3CommsLogger* on page 3-616.
- *3.6.3 SchedulerInterface* on page 3-616.
- *3.6.4 SchedulerThread* on page 3-617.
- *3.6.5 SchedulerThreadEvent* on page 3-617.

3.6.1 AsyncSignal

Allows to cleanly schedule events from non-simulation threads onto the simulation thread. This model is written in C++.

Table 3-236 Ports

Name	Protocol	Type	Description
async_callback	AsyncSignalCallback	Master	This port emits a call to signal() on the simulation thread asyncronously after async_control.signal() has been called.
async_control	AsyncSignalControl	Slave	Non-simulation threads call signal() on this port in order to schedule an event: a call to async_callback.signal() on the simulation thread.

3.6.2 GICv3CommsLogger

Traces GICv3Comms activity. This model is written in LISA+.

GICv3CommsLogger contains the following CADI targets:

- GICv3CommsLogger

GICv3CommsLogger contains the following MTI components:

- *4.74 GICv3CommsLogger - trace* on page 4-2490

Table 3-237 Ports

Name	Protocol	Type	Description
to_cpu	*GICv3Comms on page 2-84*	Master	To connect to CPU.
to_gic	*GICv3Comms on page 2-84*	Slave	To connect to GIC.

Table 3-238 Parameters for GICv3CommsLogger

Name	Type	Default value	Description
verbose	bool	0x0	Print tracing information to attached debugger in addition to via MTI

3.6.3 SchedulerInterface

A SchedulerInterface instance allows access to the Fast Models scheduler. This model is written in LISA+.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SchedulerInterface contains the following CADI targets:

- SchedulerInterface

Table 3-239 Ports

Name	Protocol	Type	Description
clk_in	ClockSignal	Slave	Clock frequency for waitTicks() function.
control	SchedulerInterfaceControl	Slave	Scheduler interface. Allows to: - wait for time

3.6.4 SchedulerThread

A SchedulerThread instance represents a co-routine thread in the simulation. This model is written in LISA+.

SchedulerThread contains the following CADI targets:

- SchedulerThread

Table 3-240 Ports

Name	Protocol	Type	Description
clk_in	ClockSignal	Slave	Clock frequency for waitTicks() function.
control	SchedulerThreadControl	Slave	SchedulerThread control. Masters use this to: - control the thread (wait etc) - implement the actual thread function threadProc()

3.6.5 SchedulerThreadEvent

A SchedulerThreadEvent instance is an auto-reset boolean condition variable other threads can wait on. This model is written in LISA+.

SchedulerThreadEvent contains the following CADI targets:

- SchedulerThreadEvent

Table 3-241 Ports

Name	Protocol	Type	Description
control	SchedulerThreadEventControl	Slave	SchedulerThreadEvent control. Masters use this to: - wait for this event - notify waiters that the event happened

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

3.7 Peripheral components

This section describes the Peripheral components.

This section contains the following subsections:

- *3.7.1 AudioOut_File* on page 3-618.
- *3.7.2 AudioOut_SDL* on page 3-619.
- *3.7.3 Base_PowerController* on page 3-619.
- *3.7.4 DebugAccessPort* on page 3-620.
- *3.7.5 DebugROM* on page 3-620.
- *3.7.6 DualClusterSystemConfigurationBlock* on page 3-621.
- *3.7.7 DummyAPB* on page 3-623.
- *3.7.8 ElfLoader* on page 3-623.
- *3.7.9 FlashLoader* on page 3-624.
- *3.7.10 GICv3CommsPVBus* on page 3-624.
- *3.7.11 GICv3ProtocolChecker* on page 3-625.
- *3.7.12 GUIPoll* on page 3-625.
- *3.7.13 HostBridge* on page 3-625.
- *3.7.14 HostSerialInterface* on page 3-626.
- *3.7.15 IntelStrataFlashJ3* on page 3-627.
- *3.7.16 MemoryMappedGenericTimer* on page 3-629.
- *3.7.17 MemoryMappedGenericWatchdog* on page 3-629.
- *3.7.18 NonVolatileCounter* on page 3-630.
- *3.7.19 PPUMultiThreadModeSwitch* on page 3-630.
- *3.7.20 PS2Keyboard* on page 3-631.
- *3.7.21 PS2Mouse* on page 3-631.
- *3.7.22 PVBusGICv3Comms* on page 3-632.
- *3.7.23 PVMetaDataController* on page 3-632.
- *3.7.24 RAMDevice* on page 3-632.
- *3.7.25 ROM* on page 3-633.
- *3.7.26 RandomNumberGenerator* on page 3-633.
- *3.7.27 RealTimeLimiter* on page 3-634.
- *3.7.28 RealtimeClockTimer* on page 3-634.
- *3.7.29 RemapDecoder* on page 3-634.
- *3.7.30 RootKeyStorage* on page 3-635.
- *3.7.31 SerialCrossover* on page 3-636.
- *3.7.32 TelnetTerminal* on page 3-636.
- *3.7.33 UnusedPrimeCell* on page 3-637.
- *3.7.34 VirtioBlockDevice* on page 3-637.
- *3.7.35 VirtioNetMMIO* on page 3-638.
- *3.7.36 VirtioP9Device* on page 3-639.
- *3.7.37 VirtioPCIBlockDevice* on page 3-641.
- *3.7.38 VirtualEthernetCrossover* on page 3-641.
- *3.7.39 VirtualEthernetHub3* on page 3-641.
- *3.7.40 VisEventRecorder* on page 3-642.
- *3.7.41 WarningMemory* on page 3-642.
- *3.7.42 v8EmbeddedCrossTrigger_Matrix* on page 3-643.

3.7.1 AudioOut_File

File based Audio Output for PL041_AACI. This model is written in LISA+.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

AudioOut_File contains the following CADI targets:

- AudioOut_File

About AudioOut_File

This component implements an audio output that is suitable for use with the PL041_AACI component. It writes raw 16-bit 48KHz stereo audio data to a user-specified file.

Arm expects this component to have little effect on the performance of PV systems. AudioOut_File drains audio data at the rate that would be expected by software running in the simulation.

Table 3-242 Ports

Name	Protocol	Type	Description
audio	*AudioControl* on page 2-71	Slave	Audio input for a connection to a component such as the PL041_AACI.

Table 3-243 Parameters for AudioOut_File

Name	Type	Default value	Description
fname	string	""	Filename

3.7.2 AudioOut_SDL

SDL based Audio Output for PL041_AACI. This model is written in LISA+.

AudioOut_SDL contains the following CADI targets:

- AudioOut_SDL

About AudioOut_SDL

AudioOut_SDL outputs audio using the host features of the Simple DirectMedia Layer library.

This component results in SDL audio callbacks and might have a small impact on PV systems containing the component. It attempts to drain audio data at whatever rate is required to maintain smooth sound playback on the host PC. This might not match the data rate expected by applications running on the simulation.

Table 3-244 Ports

Name	Protocol	Type	Description
audio	*AudioControl* on page 2-71	Slave	Audio input for a connection to a component such as the PL041_AACI.

3.7.3 Base_PowerController

Base Platforms Power Controller. This model is written in LISA+.

Base_PowerController contains the following CADI targets:

- Base_PowerController

Base_PowerController contains the following MTI components:

- *4.50 Base_PowerController - trace* on page 4-2403
- *4.107 PVBusSlave - trace* on page 4-2760

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-245 Ports

Name	Protocol	Type	Description
cpuporeset[16]	*Signal* on page 2-85	Master	-
dbgnopwrdwn[16]	*Signal* on page 2-85	Slave	-
l2reset[4]	*Signal* on page 2-85	Master	-
pchannel_m[16]	*PChannel* on page 2-80	Master	-
pvbus_s	PVBus	Slave	-
standbywfi[16]	*Signal* on page 2-85	Slave	-
standbywfil2[4]	*Signal* on page 2-85	Slave	-
system_reset	*Signal* on page 2-85	Master	-
system_reset_req	*Signal* on page 2-85	Slave	-
wakerequest[16]	*Signal* on page 2-85	Slave	-

Table 3-246 Parameters for Base_PowerController

Name	Type	Default value	Description
Affinity-shifted	bool	0x0	Are the CPU affinities shifted ? true if it is one of the APACHE cores otherwise false
CPU-affinities	string	"0.0.0.0"	Definition of which cores are attached to the control pins, as a comma separated list of affinity dotted quads
CPU-available-mask	int	-0x1	One bit per entry in CPU-affinities list, set zero if a CPU is wired up but actually not available
startup	string	"0.0.0.*"	Comma-separated list of cores (wildcards allowed) to be powered up at startup or system reset
use_pchannel_for_threads	bool	0x0	Set this to true if the pchannel is connected to cpus with thread support.

3.7.4 DebugAccessPort

This model is written in C++.

Table 3-247 Ports

Name	Protocol	Type	Description
ap_pvbus_m[2]	PVBus	Master	-
clock	ClockSignal	Slave	-
paddrdbg31	*Signal* on page 2-85	Master	-

3.7.5 DebugROM

This model is written in C++.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-248 Ports

Name	Protocol	Type	Description
paddrdbg31	*Signal* on page 2-85	Master	-
pvbus_s	PVBus	Slave	-

3.7.6 DualClusterSystemConfigurationBlock

Dual Cluster System Configuration Block. This model is written in LISA+.

DualClusterSystemConfigurationBlock contains the following CADI targets:

- DualClusterSystemConfigurationBlock

DualClusterSystemConfigurationBlock contains the following MTI components:

- *4.69 DualClusterSystemConfigurationBlock - trace* on page 4-2464
- *4.107 PVBusSlave - trace* on page 4-2760

Table 3-249 Ports

Name	Protocol	Type	Description
clk_in	ClockSignal	Slave	-
cluster0_cfgend[4]	*Signal* on page 2-85	Master	-
cluster0_cfgte[4]	*Signal* on page 2-85	Master	-
cluster0_clusterid	*Value* on page 2-85	Master	-
cluster0_corereset[4]	*Signal* on page 2-85	Master	-
cluster0_cpuporeset[4]	*Signal* on page 2-85	Master	-
cluster0_cxreset[4]	*Signal* on page 2-85	Master	-
cluster0_eventi	*Signal* on page 2-85	Peer	-
cluster0_evento	*Signal* on page 2-85	Peer	-
cluster0_iminlen	*Signal* on page 2-85	Master	-
cluster0_l2reset	*Signal* on page 2-85	Master	-
cluster0_standbywfi[4]	*Signal* on page 2-85	Slave	-
cluster0_vinithi[4]	*Signal* on page 2-85	Master	-
cluster1_cfgend[4]	*Signal* on page 2-85	Master	-
cluster1_clusterid	*Value* on page 2-85	Master	-
cluster1_corereset[4]	*Signal* on page 2-85	Master	-
cluster1_cpuporeset[4]	*Signal* on page 2-85	Master	-
cluster1_eventi	*Signal* on page 2-85	Peer	-
cluster1_evento	*Signal* on page 2-85	Peer	-
cluster1_scureset	*Signal* on page 2-85	Master	-
cluster1_standbywfi[4]	*Signal* on page 2-85	Slave	-
cluster1_teinit[4]	*Signal* on page 2-85	Master	-
cluster1_vinithi[4]	*Signal* on page 2-85	Master	-

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-249 Ports (continued)

Name	Protocol	Type	Description
daughter_leds_state	*ValueState* on page 2-85	Master	-
daughter_user_switches	*ValueState* on page 2-85	Master	-
intgen[128]	*Signal* on page 2-85	Master	-
periphbase	*Value_64* on page 2-85	Master	-
periphbase_32	*Value* on page 2-85	Master	-
pvbus	PVBus	Slave	-
system_reset	*Signal* on page 2-85	Master	-
vgic_configuration_port	v7_VGIC_Configuration_Protocol	Master	-

Table 3-250 Parameters for DualClusterSystemConfigurationBlock

Name	Type	Default value	Description
CFG_ACTIVECLUSTER	int	0x1	Select which cluster will come out of reset coming out of power-on: bit[0] for primary cluster (Cortex-A15), bit[1] for secondary cluster (Cortex-A7). Value 0 is not allowed as it will hold both clusters in reset indefinitely!
Cluster0IdOnPOReset	int	0x0	ClusterId for primary cluster (Cortex-A15) on power-on reset
Cluster1IdOnPOReset	int	0x1	ClusterId for secondary cluster (Cortex-A7) on power-on reset
DCSCB_PERIPHBASE	int	0x1e000000	PERIPHBASE
DCS_AID	int	0x0	DCS_AID is the Auxiliary ID Register
DCS_ID	int	0x41120000	The value returned by the DCS_ID register.
DCS_ID_BUILD_NUMBER	int	0x1	DCS_ID build number
DCS_LEDS	int	0x0	DCS_LEDS represents eight LEDs on the board that form an 8-bit value that can be r/w from the Dual Cluster System Configuration Block
DCS_SW	int	0x0	DCS_SW represents eight switches on the board that form an 8-bit value that can be read from the Dual Cluster System Configuration Block
FlipVGICWiringForCluster0AndCluster1	bool	0x0	Flip the VGIC wiring round for cluster0 and cluster1. With this false, then cpu0 of cluster0 is cpu interface 0 on the VGIC. If this is true then cpu0 of cluster1 becomes cpu interface 0 on the VGIC.
INTGEN_INTS	int	0x3	Number of custom IRQs controlled by interrupt generator is INTGEN_INTS * 32 + 32.
NumberOfCoresInCluster0	int	0x0	The number of cores in the primary cluster
NumberOfCoresInCluster1	int	0x0	The number of cores in the secondary cluster
ResetValueOfDaughterUserSwitches	int	0x0	Reset value of the user switches on the daughterboard
stop_on_sequence_id	int	0x0	If non-zero the sequence_id of the SW trace mechanism on which to halt the simulator

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

3.7.7 DummyAPB

DummyAPB. This model is written in LISA+.

DummyAPB contains the following CADI targets:

- DummyAPB

DummyAPB contains the following MTI components:

- *4.107 PVBusSlave - trace* on page 4-2760

About DummyAPB

Use this dummy RAZ/WI APB device component to ensure that software does not receive aborts for accesses to devices that should be part of the system, but are not modeled.

For validation purposes it is useful to have dummy devices that are mostly RAZ/WI but return the correct value when you read ID registers. You can do that with this component in the following ways:

- Specify `periphid_24` for peripherals that follow the ARM pattern of having 12 ID registers at the top of an APB frame. For example:

  ```
  periphid_24="04000000c2b00b000df005b1"
  ```

  You need also to set `periph_framesize` to 4 or 64, depending on whether the peripheral has its registers in a 4KB or 64KB frame.
- Give a space-separated list of *offset:value* pairs in the `periphid_generic` parameter to define read-only values from particular offsets. For example:

  ```
  periphid_generic="000:02468ace 1fc:13579bdf"
  ```

  The number of hex digits used to specify the address is used to define the width of the address mask used. For example, `BC:02468ace` returns `02468ace` at reads from any address ending `BC`.
- Give a space-separated list of *offset:default-value* pairs in the `ram_generic` parameter to construct RAM. That is, the register at the relevant offset returns the default-value, but if changed, it returns the value that it is changed to.

Table 3-251 Ports

Name	Protocol	Type	Description
`pvbus_s`	`PVBus`	Slave	Bus slave interface.

Table 3-252 Parameters for DummyAPB

Name	Type	Default value	Description
`fail`	`bool`	`0x0`	Abort all accesses
`failmsg`	`string`	`""`	String to print when 'fail'=true and access occured
`periph_framesize`	`int`	`-0x1`	Size of frame (4/64, indicating if ID is at xFD0 or xFFD0)
`periphid_24`	`string`	`""`	24 hex digits for the 12 bytes of peripheral ID
`periphid_generic`	`string`	`""`	Set of space-separated offset:value pairs for dwords of ID
`ram_generic`	`string`	`""`	Set of space-seperated offset:default pairs for writable dwords
`warn_once`	`bool`	`0x1`	Warn once for the invalid read and write access.

3.7.8 ElfLoader

ELF loader component. This model is written in LISA+.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ElfLoader contains the following CADI targets:

- ElfLoader

ElfLoader contains the following MTI components:

- *4.106 PVBusMaster - trace* on page 4-2759

About ElfLoader

ElfLoader provides an alternative method of loading ELF files into the system. It can load files in any of the following formats, or in gzip-compressed versions of them:

- ELF.
- Motorola S-Record.
- Intel-Hex.
- Verilog-Hex, in the format:

```
@<address_in_hex> <byte_in_hex>
```

Table 3-253 Ports

Name	Protocol	Type	Description
pvbus_m	PVBus	Master	Master port for all memory accesses.
start_address	*Value_64* on page 2-85	Master	Provides a value reflecting the entry point of the last ELF image to be loaded.

Table 3-254 Parameters for ElfLoader

Name	Type	Default value	Description
elf	string	""	ELF file
lfile	string	""	load file for large address mapping
ns_copy	bool	0x1	copy whole file to NS memory space

3.7.9 FlashLoader

A device that can preload a gzipped image into flash at startup. This model is written in C++.

FlashLoader contains the following CADI targets:

- FlashLoader

About FlashLoader

This component complements the IntelStrataFlashJ3 component by providing a means to initialize the contents of up to four Flash components in sequence from a single host flash image file.

Table 3-255 Ports

Name	Protocol	Type	Description
flash_device0	*FlashLoaderPort* on page 2-71	Master	Used to program a flash device.
flash_device1	*FlashLoaderPort* on page 2-71	Master	Used to program a flash device.
flash_device2	*FlashLoaderPort* on page 2-71	Master	Used to program a flash device.
flash_device3	*FlashLoaderPort* on page 2-71	Master	Used to program a flash device.

3.7.10 GICv3CommsPVBus

GICv3 Component for conversion between GICv3Comms protocol and PVBus. This model is written in C++.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

GICv3CommsPVBus contains the following CADI targets:

- GICv3CommsPVBus

GICv3CommsPVBus contains the following MTI components:

- *4.75 GICv3CommsPVBus - trace* on page 4-2494
- *4.107 PVBusSlave - trace* on page 4-2760

Table 3-256 Ports

Name	Protocol	Type	Description
axi_master_id_s[256]	*Value* on page 2-85	Slave	-
distributor_s[256]	*GICv3Comms* on page 2-84	Slave	-
pvbus_m	PVBus	Master	-
pvbus_s	PVBus	Slave	-

3.7.11 GICv3ProtocolChecker

GICv3 Component for command protocol checking. This model is written in C++.

GICv3ProtocolChecker contains the following CADI targets:

- GICv3ProtocolChecker

GICv3ProtocolChecker contains the following MTI components:

- *4.79 GICv3ProtocolChecker - trace* on page 4-2526

Table 3-257 Ports

Name	Protocol	Type	Description
cpu_comms	*GICv3Comms* on page 2-84	Master	Master GICv3Comms port.
gicv3_comms	*GICv3Comms* on page 2-84	Slave	Slave GICv3Comms port.

3.7.12 GUIPoll

Component providing a real-time periodic callback for GUI refresh. This model is written in C++.

About GUIPoll

Arm intends this component to be used as a subcomponent inside a LISA-based visualization component.

You can configure the period at which the GUIPoll component runs in milliseconds of real time, not simulation time. The component produces a `gui_callback()` signal at approximately this period, even when the simulation is paused.

You must implement the slave side of the `gui_callback()` signal in a LISA-based visualization component. Use this event to invoke the `Visualisation::poll()` method to keep the GUI updated.

Table 3-258 Ports

Name	Protocol	Type	Description
gui_callback	*GUIPollCallback* on page 2-71	Master	Sends callback requests to the visualization component.

3.7.13 HostBridge

Host Socket Interface Component. This model is written in C++.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

HostBridge contains the following CADI targets:

- HostBridge

About HostBridge

This component acts as a networking proxy for target NIC device models, to forward and receive ethernet packets to and from the host. Two kinds of proxy backend are integrated into this component:

- A host TAP/TUN-like network device, which is an ordinary TAP or MacVTap. This is the default.
- User mode networking, which emulates a built-in IP router and DHCP server to route traffic using the host user mode socket layer. To enable user mode networking, set the `userNetworking` parameter to true.

  For more information see *1.11.1 User mode networking* on page 1-38.

HostBridge requires the following initialization sequence:

```
hostbridge.state.setValue(HostBridge::STATUS);
hostbridge.state.setValue(HostBridge::S_UP);
```

To enable tracing of user-mode networking, which can help to debug networking issues, set the `FASTSIM_USERNET_DUMP` environment variable to any or all of the following values:

```
arpin,arpout,udpin,udpout,etherin,etherout,ipv4in,ipv4out,ipv4fragin,ipv4fragout,tcpin,tcpout
```

Table 3-259 Ports

Name	Protocol	Type	Description
`eth`	*`VirtualEthernet` on page 2-79*	Slave	-
`state`	`ValueState_64`	Slave	-

Table 3-260 Parameters for HostBridge

Name	Type	Default value	Description
`interfaceName`	`string`	""	Host Interface
`userNetOptions`	`string`	""	Control options for UserNet TCP/IP (for internal use only, please do not use)
`userNetPorts`	`string`	""	Listening ports to expose in user-mode networking
`userNetSubnet`	`string`	"172.20.51.0/24"	Virtual subnet for user-mode networking
`userNetworking`	`bool`	`0x0`	Enable user-mode networking

Related tasks

Configuring the networking environment for Microsoft Windows on page 1-39

Configuring the networking environment for Linux on page 1-44

Related information

Fast Models User Guide

3.7.14 HostSerialInterface

Component which provides access to the host serial interface. This model is written in LISA+.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

HostSerialInterface contains the following CADI targets:

- HostSerialInterface

Table 3-261 Ports

Name	Protocol	Type	Description
SerialData	*SerialData on page 2-77*	Slave	Serial data connection to export to host machine.

Table 3-262 Parameters for HostSerialInterface

Name	Type	Default value	Description
baud_rate	int	0x0	Baud rate override
device	string	"/dev/ttyS0"	HW device to use

3.7.15 IntelStrataFlashJ3

Intel Strata Flash J3 LISA+ model. This model is written in LISA+.

IntelStrataFlashJ3 contains the following CADI targets:

- IntelStrataFlashJ3

IntelStrataFlashJ3 contains the following MTI components:

- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760

About IntelStrataFlashJ3

This component is an efficient implementation of a NOR flash memory type device, an Intel StrataFlash Memory (J3).

In normal usage, the device acts as *Read Only Memory* (ROM) whose contents can be determined either by programming using the flashloader port or by using standard flash programming software running on the model, such as the Arm Firmware Suite.

The implementation of this component is approximately that of the Intel part in the VE development board. The component is effectively organized as a bank of two 16-bit Intel Flash components forming a 32-bit component that can be read or programmed in parallel. The component supports all hardware behavior except for:

- Protection register.
- Enhanced configuration register.
- Unique device identifier.
- One time programmable cells.
- Suspend/resume, which is silently ignored.
- Status interrupt line.

All block operations are atomic. This means that the status register state machine status bit always reads 1, ready.

In normal operation, this component has no user-visible registers, but you can read from it as if it is memory.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Programming it or changing the configuration requires a sequence of special write operations, see general flash programming documentation. The component supports Common Flash Interface query operations, which allow drivers to determine the properties of the flash memory.

Note

The model interprets all writes as requests to the programming state machine, and there are many state-machine states that do not support subsequent reads and return `0xdeaddead` for them. Therefore, when simulating a ROM, use the `trapwrite=true` option.

Use the `diagnostics` parameter to select the level of diagnostic output:

Level 0
None.

Level 1
Report probable driver error operations:
- Unaligned operations that fault.
- Accesses that the state machine does not expect.
- Transitions of the state machine to unknown states.
- Writes to locked blocks and illegal lock commands.

Level 2

Report unimplemented and therefore ignored operations, and log lock commands.

Level 3
Warn if a flash write attempts to set bits (the write works if `unphysical_writes=true`).

Level 4
Log every read and write.

Table 3-263 Ports

Name	Protocol	Type	Description
`flashloader`	*`FlashLoaderPort` on page 2-71*	Slave	Can be used to load code/data into the flash.
`mem_port`	*`PVDevice` on page 2-77*	Slave	PVDevice port that is connect PVBusSlave port.
`pvbus`	`PVBus`	Slave	This is where the address and the data comes from the bus.

Table 3-264 Parameters for IntelStrataFlashJ3

Name	Type	Default value	Description
`diagnostics`	`int`	`0x0`	Diagnostic level
`enable_read_status_logic`	`bool`	`0x0`	Enables logic to handle the status register reads as per the '3 Volt Intel StrataFlash Memory' specification
`model_blocklock`	`bool`	`0x0`	Model per-block locking and set all the blocks to locked state on reset
`size`	`int`	`0x40000`	Memory Size
`trapwrite`	`bool`	`0x0`	Generate abort on write
`unphysical_writes`	`bool`	`0x1`	Writes to flash are overwrite not AND

Related information

Intel Download Center, Intel StrataFlash Memory (J3) datasheet

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

3.7.16 MemoryMappedGenericTimer

ARM Generic Timer. This model is written in LISA+.

MemoryMappedGenericTimer contains the following CADI targets:

- MemoryMappedGenericTimer

MemoryMappedGenericTimer contains the following MTI components:

- *4.107 PVBusSlave - trace* on page 4-2760

Table 3-265 Ports

Name	Protocol	Type	Description
cntpsirq[8]	*Signal* on page 2-85	Master	-
cntvalueb	*CounterInterface* on page 2-84	Slave	For connection to MemoryMappedCounterModule.
pvbus_base_s[8]	PVBus	Slave	-
pvbus_ctlbase_s	PVBus	Slave	-
pvbus_el0base_s[8]	PVBus	Slave	-
timer_reset	*Signal* on page 2-85	Slave	Reset.

Table 3-266 Parameters for MemoryMappedGenericTimer

Name	Type	Default value	Description
bypass_ctlbase	bool	0x0	Bypass CNTBase Access Control. Enable if only timer frame feature is required without CNTBase access control
cntel0acr_implemented	int	0x0	A bit-field of 8 bits, where bit {n} enables CNTEL0ACR for timer frame {n}
diagnostics	int	0x0	Diagnostics
frame_security	string	""	Hard-wired/configurable security for frames (N/S/X, one character per timer frame)
num_timers	int	0x1	Number of timer frames

3.7.17 MemoryMappedGenericWatchdog

ARM Generic Watchdog. This model is written in LISA+.

MemoryMappedGenericWatchdog contains the following CADI targets:

- MemoryMappedGenericWatchdog

MemoryMappedGenericWatchdog contains the following MTI components:

- *4.107 PVBusSlave - trace* on page 4-2760

About MemoryMappedGenericWatchdog

This is a high-level watchdog that generates two interrupts rather than an interrupt followed by a reset.

Table 3-267 Ports

Name	Protocol	Type	Description
WS0	*Signal* on page 2-85	Master	Set when watchdog expired for the first time.
WS1	*Signal* on page 2-85	Master	Set when watchdog expired for the second time.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-267 Ports (continued)

Name	Protocol	Type	Description
cntvalueb	*CounterInterface* on page 2-84	Slave	For connection to MemoryMappedCounterModule.
ctl_pvbus_s	PVBus	Slave	Access to control frame.
ref_pvbus_s	PVBus	Slave	Access to refresh frame.
reset_in	*Signal* on page 2-85	Slave	Reset.

Table 3-268 Parameters for MemoryMappedGenericWatchdog

Name	Type	Default value	Description
NONSECURE	bool	0x0	Non-Secure
diagnostics	int	0x0	Diagnostics
product_id	int	0x0	Product Identifier

3.7.18 NonVolatileCounter

Trusted Non-Volatile Counter unit. This model is written in LISA+.

NonVolatileCounter contains the following CADI targets:

- NonVolatileCounter

NonVolatileCounter contains the following MTI components:

- *4.93 NonVolatileCounter - trace* on page 4-2736
- *4.107 PVBusSlave - trace* on page 4-2760

Table 3-269 Ports

Name	Protocol	Type	Description
pvbus_s	PVBus	Slave	Bus slave interface.

Table 3-270 Parameters for NonVolatileCounter

Name	Type	Default value	Description
diagnostics	int	0x0	Diagnostics
rst_non_tz_fw_cnt	int	0x0	Value of NON_TZ_FW_CNT at reset
rst_tz_fw_cnt	int	0x0	Value of TZ_FW_CNT at reset
secure	bool	0x1	Instantiate model as Secure (1) or NS (0)
version	string	"r0"	Version of the model functionality. Valid values are r0 and r1.

3.7.19 PPUMultiThreadModeSwitch

PPU mode switch between single-thread mode and multi-thread mode. Support up to 8 cores and thread number per core is no more than 2. This model is written in C++.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-271 Ports

Name	Protocol	Type	Description
pchannel_from_ppu_s[8]	*PChannel* on page 2-80	Slave	-
pchannel_to_cpu_m[8]	*PChannel* on page 2-80	Master	-
wakerequest_from_gic_s[16]	*Signal* on page 2-85	Slave	-
wakerequest_to_ppu_m[8]	*Signal* on page 2-85	Master	-

3.7.20 PS2Keyboard

Interface component, which takes the keypress/release signals from the Visualisation component and translates them into clocked PS2Data signals which can be routed to a PL050_KMI component. This model is written in LISA+.

PS2Keyboard contains the following CADI targets:

- PS2Keyboard

About PS2Keyboard

This component translates a stream of key press information into appropriate PS/2 serial data. The key press data stream must be provided by another component such as a visualization component.

Table 3-272 Ports

Name	Protocol	Type	Description
clk_in	ClockSignal	Slave	Clock signal to rate at which PS2Data signals are generated.
keyboard	*KeyboardStatus* on page 2-72	Slave	Receives keyboard input from, for example, the Visualisation component.
ps2	*PS2Data* on page 2-74	Master	Connection to the PS/2 controller, for example, the PL050_KMI.

3.7.21 PS2Mouse

Interface component, which takes the keypress/release signals from the Visualisation component and translates them into clocked PS2Data signals which can be routed to a PL050_KMI component. This model is written in LISA+.

PS2Mouse contains the following CADI targets:

- PS2Mouse

About PS2Mouse

This component implements the PS/2 register interface of a PS/2 style mouse. The mouse movement and button press data must come from another component such as a visualization component.

Table 3-273 Ports

Name	Protocol	Type	Description
clk_in	ClockSignal	Slave	Clock signal to rate at which PS2Data signals are generated.
mouse	*MouseStatus* on page 2-74	Slave	Receives keyboard input from, for example, the Visualisation component.
ps2	*PS2Data* on page 2-74	Master	Connection to the PS/2 controller, for example the PL050_KMI.

Related references

1.16 Visualisation Library on page 1-53

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

3.7.22 PVBusGICv3Comms

GICv3 Component for conversion between GICv3Comms protocol and PVBus. This model is written in C++.

PVBusGICv3Comms contains the following CADI targets:

- PVBusGICv3Comms

PVBusGICv3Comms contains the following MTI components:

- *4.103 PVBusGICv3Comms - trace* on page 4-2754
- *4.107 PVBusSlave - trace* on page 4-2760

Table 3-274 Ports

Name	Protocol	Type	Description
axi_master_id_m[256]	*Value* on page 2-85	Master	-
distributor_m[256]	*GICv3Comms* on page 2-84	Master	-
pvbus_m	PVBus	Master	-
pvbus_s	PVBus	Slave	-

3.7.23 PVMetaDataController

A simulation-only (not in hardware) component that can service metadata requests for transactions on PVBus. This model is written in C++.

Table 3-275 Ports

Name	Protocol	Type	Description
pvbus_m	PVBus	Master	-
pvbus_s	PVBus	Slave	-

3.7.24 RAMDevice

RAM device, can be dynamic or static ram. This model is written in LISA+.

RAMDevice contains the following CADI targets:

- RAMDevice

RAMDevice contains the following MTI components:

- *4.107 PVBusSlave - trace* on page 4-2760
- *4.110 RAMDevice - trace* on page 4-2769

Note

As a generic device, this component does not have a hardware revision code.

Table 3-276 Ports

Name	Protocol	Type	Description
pvbus	PVBus	Slave	Bus slave interface.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-277 Parameters for RAMDevice

Name	Type	Default value	Description
enable_atomic_ops	bool	0x0	Supports Atomic Operations
fill1	int	0xdfdfdfcf	Fill pattern 1, initialise memory at start of simulation with alternating fill1, fill2 pattern
fill2	int	0xcfdfdfdf	Fill pattern 2, initialise memory at start of simulation with alternating fill1, fill2 pattern
size	int	0x100000000	Memory Size

3.7.25 ROM

Simple ROM device. This model is written in LISA+.

ROM contains the following CADI targets:

- ROM

ROM contains the following MTI components:

- *4.107 PVBusSlave - trace* on page 4-2760

Table 3-278 Ports

Name	Protocol	Type	Description
pvbus	PVBus	Slave	-

Table 3-279 Parameters for ROM

Name	Type	Default value	Description
abort_writes	bool	0x0	Abort writes instead of ignoring them
log2_size	int	0x14	Log2 size (bytes) e.g. 20 is 1 MiB
raw_image	string	""	Raw image file to load at init time

3.7.26 RandomNumberGenerator

Random Number Generator unit. This model is written in LISA+.

RandomNumberGenerator contains the following CADI targets:

- RandomNumberGenerator

RandomNumberGenerator contains the following MTI components:

- *4.107 PVBusSlave - trace* on page 4-2760

Table 3-280 Ports

Name	Protocol	Type	Description
RNG_intr	*Signal* on page 2-85	Master	Interrupt output.
pvbus_s	PVBus	Slave	Bus slave interface.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-281 Parameters for RandomNumberGenerator

Name	Type	Default value	Description
diagnostics	int	0x0	Diagnostics
seed	int	0x0	Random number seed

3.7.27 RealTimeLimiter

Real Time Limiter. This model is written in LISA+.

RealTimeLimiter contains the following CADI targets:

- ClockDivider
- RealTimeLimiter

RealTimeLimiter contains the following MTI components:

- *4.60 ClockDivider - trace* on page 4-2438

Table 3-282 Ports

Name	Protocol	Type	Description
clk_in	ClockSignal	Slave	Clock input.

Table 3-283 Parameters for RealTimeLimiter

Name	Type	Default value	Description
ENABLE	bool	0x0	Rate limit simulation
RELATIVE_SPEED	int	0x64	Rate limit to at most this percentage of real time (100: limit to wall clock rate)

3.7.28 RealtimeClockTimer

Host Time Based Timer Module for Generic Timers. This model is written in C++.

Table 3-284 Ports

Name	Protocol	Type	Description
set_frequency	*Value_64* on page 2-85	Slave	-
timer_callback	*TimerCallback64* on page 2-65	Master	-
timer_control	*TimerControl64* on page 2-66	Slave	-

3.7.29 RemapDecoder

The component that provides support for dynamically remappable regions of memory. This model is written in LISA+.

RemapDecoder contains the following CADI targets:

- RemapDecoder
- TZSwitch

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

RemapDecoder contains the following MTI components:

- *4.105 PVBusMapper - trace* on page 4-2758

Table 3-285 Ports

Name	Protocol	Type	Description
control	TZSwitchControl	Broadcast	-
input	PVBus	Slave	Incoming bus transactions (connected straight to TZSwitch).
output_remap_clear	PVBus	Master	Outgoing bus transactions when remap is clear.
output_remap_set	PVBus	Master	Outgoing bus transactions when remap is set.
remap	*StateSignal* on page 2-85	Slave	Remapping control.

Table 3-286 Parameters for TZSwitch

Name	Type	Default value	Description
normal	int	0x2	Normal Port
secure	int	0x1	Secure Port

3.7.30 RootKeyStorage

Trusted Root-Key Storage unit. This model is written in LISA+.

RootKeyStorage contains the following CADI targets:

- RootKeyStorage

RootKeyStorage contains the following MTI components:

- *4.107 PVBusSlave - trace* on page 4-2760

Table 3-287 Ports

Name	Protocol	Type	Description
pvbus_s	PVBus	Slave	Bus slave interface.

Table 3-288 Parameters for RootKeyStorage

Name	Type	Default value	Description
diagnostics	int	0x0	Diagnostics
hw_unique_key	string	"00000000 00000000 00000000 00000000"	Hardware Unique Key (128-bit, 4 hex words)
hw_unique_key_hex	string	""	Hardware Unique Key (128-bit, little-endian hex byte stream)
private_key	string	"00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000"	Private Endorsement Key (256-bit, 8 hex words)
private_key_hex	string	""	Private Key (256-bit, little-endian hex byte stream)
public_key	string	"00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000"	Public Key (256-bit, 8 hex words)
public_key_hex	string	""	Public Key (256-bit, little-endian hex byte stream)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-288 Parameters for RootKeyStorage (continued)

Name	Type	Default value	Description
ss_key	string	"00000000 00000000 00000000 00000000"	Secret Symmetric Key (128-bit, 4 hex words)
ss_key_hex	string	""	Secret Symmetric Key (128-bit, little-endian hex byte stream)
version	string	"r1"	Version of the model functionality. Valid values are r0 and r1.

3.7.31 SerialCrossover

Implement an equivalent to a null-modem cable, swapping over serial transmit and receive signals. This model is written in C++.

About SerialCrossover

This component implements two SerialData slave ports and can connect two SerialData master ports, such as from PL011_Uart components. Data received on one port is buffered in a FIFO until it is read from the other port. Signals received on one port are latched and available to be read by the other port.

Table 3-289 Ports

Name	Protocol	Type	Description
port_a	*SerialData* on page 2-77	Slave	Slave port for connecting to a SerialData master.
port_b	*SerialData* on page 2-77	Slave	Slave port for connecting to a SerialData master.

3.7.32 TelnetTerminal

A host interface onto a serial port: exposes the two way serial data channel over a TCP/IP interface, and automatically opens a telnet application connected to the network socket, unless a user application connects first. This model is written in C++.

TelnetTerminal contains the following CADI targets:

- TelnetTerminal

About TelnetTerminal

When the simulation is started and the TelnetTerminal component is enabled, the component opens a server (listening) socket on a TCP/IP port. This is port 5000 by default.

Data written to the SerialData port is transmitted over the network socket. When data becomes available on the network socket, the TelnetTerminal component buffers the data. The data can then be read from SerialData.

If there is no connection to the network socket when the first data access is made, a host telnet session is automatically started. Prior to this first access, you can connect a client of your choice to the network socket. If the connection between the TelnetTerminal component and the client is broken at any time, the port is re-opened, permitting you to make another client connection.

To set the parameters, the syntax to use in a configuration file or on the command line is:

`motherboard.terminal_x.parameter=value`

where *x* is the terminal identifier and can be 0, 1, 2, or 3.

The `mode` parameter selects the terminal initialisation mode. You can set it to either of these values:

`telnet` In telnet mode, this component supports a subset of the telnet protocol defined in RFC 854.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

`raw` In raw mode, this component does not interpret or modify the byte stream contents. This permits a debugger connection, for example, to connect a gdb client to a gdbserver running on the target operating system.

The `terminal` parameter specifies the command line used to launch a terminal application and connect to the opened TCP port. The `TelnetTerminal` component replaces the keywords `%port` and `%title`, if specified, with the opened port number and component name, respectively. After replacing `%port` and `%title`, the command line is executed verbatim.

An empty string, which is the default, launches xterm on Linux or `telnet.exe` on Windows.

Note

If you specify a non-empty string, it must include `%port`, but `%title` is optional.

For example:

```
motherboard.terminal_0.terminal="putty.exe -telnet %port localhost"
```

Table 3-290 Ports

Name	Protocol	Type	Description
`serial`	*`SerialData` on page 2-77*	Slave	Slave port for connecting to a SerialData master.

Related concepts

1.10 Using the TelnetTerminal on page 1-36

Related references

3.9.42 PL011_Uart on page 3-811

3.7.33 UnusedPrimeCell

A dummy component. It can be used to represent any unimplemented PrimeCell components. This model is written in LISA+.

UnusedPrimeCell contains the following CADI targets:

- UnusedPrimeCell

UnusedPrimeCell contains the following MTI components:

- *4.107 PVBusSlave - trace* on page 4-2760

Table 3-291 Ports

Name	Protocol	Type	Description
`pvbus`	`PVBus`	Slave	Bus slave interface.

3.7.34 VirtioBlockDevice

virtio block device. This model is written in C++.

VirtioBlockDevice contains the following CADI targets:

- VirtioBlockDevice

VirtioBlockDevice contains the following MTI components:

- *4.106 PVBusMaster - trace* on page 4-2759
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.119 VirtioBlockDevice - trace* on page 4-2868

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

About VirtioBlockDevice

VirtioBlockDevice implements a block device that can be accessed from the simulated OS if it has an appropriate driver. Similarly to the VirtioP9Device, this component is targeted primarily at Linux, which has a built-in Virtio block driver. VirtioBlockDevice allows you to use a file on the host that you specify using the `image_path` parameter, as a hard drive in the simulated OS.

Unlike the VirtioP9Device, you should not need to carry out any special setup to use VirtioBlockDevice on VE or Base platforms, because it is usually already included in the device trees. Set the `image_path` parameter to point to your image, and then on Linux it is available as a block device, usually `/dev/vda`, which you then use like any other hard drive.

Table 3-292 Ports

Name	Protocol	Type	Description
`intr`	*Signal* on page 2-85	Master	Virtio device sets interrupt to signal completion.
`pvbus`	`PVBus`	Slave	Virtio MMIO control/config/status registers.
`virtio_m`	`PVBus`	Master	Virtio device performs DMA accesses via master.

3.7.35 VirtioNetMMIO

Virtio net device over MMIO transport. This model is written in C++.

VirtioNetMMIO contains the following CADI targets:

- VirtioNetMMIO

VirtioNetMMIO contains the following MTI components:

- *4.106 PVBusMaster - trace* on page 4-2759

About VirtioNetMMIO

This is a model of a virtual Ethernet virtio device over MMIO transport, supporting both legacy and current OASIS virtio specifications. It provides much better network performance than the SMSC_91C111 component, because it features host-assisted network acceleration. This means that it can offload packet processing operations from the simulated OS on the target, to the host side. These operations include:

- Checksum computation
- TX packet segmentation
- RX packet combination

If the target simulated Linux or Linux-derived OS has an appropriate virtio net driver, Arm recommends you use VirtioNetMMIO instead of SMSC_91C111.

Unlike SMSC_91C111, which must work with an external HostBridge component, VirtioNetMMIO has a built-in HostBridge sub-component. The parameters to control the HostBridge are described in the VirtioNetMMIO parameters table, with the `hostbridge` parameter sub-namespace.

Note

To enable tracing of user-mode networking, which can help to debug networking issues, set the `FASTSIM_USERNET_DUMP` environment variable to any or all of the following values:

```
arpin,arpout,udpin,udpout,etherin,etherout,ipv4in,ipv4out,ipv4fragin,ipv4fragout,tcpin,tcpout
```

Take the following steps to set up this component in a virtual platform:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

- Use a version of Linux that contains a virtio network driver.
- Add the following option to the Linux kernel configuration:

```
CONFIG_VIRTIO_NET=y
```

- Update the device tree to include the `VirtioNetMMIO` component, or specify it on the kernel command line, for example

```
virtio_mmio.device=0x10000@0x1c150000:76
```

  The address range for both VE and Base platforms is `0x1C150000-0x1C15FFFF`. The interrupt number is 44, or IRQ 76, for both VE and Base platforms.
- Select the hostbridge that you want to use to communicate with the host in the model:

```
virtio_net.hostbridge.userNetworking=true/false (User mode or TAP/TUN networking)
```

- Configure the networking environment, as described in the related tasks at the end of this topic.

Example entries for DTS files:

- Add this entry next to the corresponding `virtio_block` or `virtio_p9` entry:

```
virtio_net@0150000 {
        compatible = "virtio,mmio";
        reg = <0x150000 0x1000>;
        interrupts = <0x2c>;
};
```

- Add this entry to the interrupt map:

```
<0 0 44 &gic 0 44 4>;
```

Table 3-293 Ports

Name	Protocol	Type	Description
`intr`	*Signal* on page 2-85	Master	Virtio device sets interrupt to signal completion.
`pvbus`	`PVBus`	Slave	Virtio MMIO control/config/status registers.
`virtio_m`	`PVBus`	Master	Virtio device performs DMA accesses via master.

Table 3-294 Parameters for VirtioNetMMIO

Name	Type	Default value	Description
`checksum`	`string`	""	For checksum-offloaded packets, if 'tx' is specified, outgoings will be checksumed by VirtioNet device; 'rx' is specified for incomings; 'all' for both
`enabled`	`bool`	`0x0`	Enable or disable device. If disabled, device can be accessed, but will not be activated
`mac_address`	`string`	""	Device MAC address, if not specified, a random MAC address is generated
`offload`	`string`	"gso, gro"	Offload TCP/UDP segmentation/receiving operations to host
`secure_accesses`	`bool`	`0x0`	Make device generate transactions with NS=0
`target_endian`	`bool`	`0x0`	Target cpu endianness(0-LE, 1-BE)
`transport`	`string`	"modern"	Choose legacy or modern virtio transport, if not specified, modern transport is used

Related tasks

Configuring the networking environment for Microsoft Windows on page 1-39

Configuring the networking environment for Linux on page 1-44

3.7.36 VirtioP9Device

virtio P9 server. This model is written in C++.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VirtioP9Device contains the following CADI targets:

- VirtioP9Device

VirtioP9Device contains the following MTI components:

- *4.106 PVBusMaster - trace* on page 4-2759
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.120 VirtioP9Device - trace* on page 4-2871

About VirtioP9Device

This component implements a subset of the Plan 9 file protocol over a virtio transport. It enables accessing a directory on the host's filesystem within Linux, or another operating system that implements the protocol, running on a platform model.

The component implements a subset of the Linux 9P2000.L protocol. It has the following limitations:

- You can mount only one host directory per instance of the component.
- It supports a subset of 9P2000.L message types: Tversion, Tlopen, Tlcreate, Tgetattr, Tsetattr, Treaddir, Tmkdir, Tattach, Twalk, Tread, Twrite, Tclunk, Tremove, Trename. On Linux hosts, it also supports: Treadlink, Tsymlink.
- On Windows hosts, it ignores Unix permissions when writing files.
- On Windows hosts, it performs a simple mapping from Windows to Unix permissions when reading.
- On Windows hosts, symbolic links appear as regular files.
- On Windows hosts, it does not perform writing, deleting, or renaming operations on a file that another process has open.

Take the following steps to set up this component:

- Use a version of Linux that supports v9fs over virtio and virtio-mmio devices.
- Update the device tree to include the VirtioP9Device component, or specify it on the kernel command-line, as shown below. The address range for both VE and Base platforms is `0x1C140000-0x1C14FFFF`.

  The interrupt number is 43, or IRQ 75, for both VE and Base platforms.
- Set the following parameter to the directory on the host that you want to mount in the model:

  VE:

  `motherboard.virtiop9device.root_path`

  Base:

  `bp.virtiop9device.root_path`
- On Linux, mount the host directory by using the following command in the model:

```
$ mount -t 9p -o trans=virtio,version=9p2000.L FM <mount point>
```

Example kernel command-line argument:

```
virtio_mmio.device=0x10000@0x1c140000:75
```

Example entry for DTS files, to add next to the corresponding `virtio_block` entry:

```
virtio_p9@0140000 {
       compatible = "virtio,mmio";
       reg = <0x0 0x1c140000 0x0 0x1000>;
       interrupts = <0x0 0x2b 0x4>;
};
```


Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-295 Ports

Name	Protocol	Type	Description
intr	*Signal* on page 2-85	Master	Virtio device sets interrupt to signal completion.
pvbus	PVBus	Slave	Virtio MMIO control/config/status registers.
virtio_m	PVBus	Master	Virtio device performs DMA accesses via master.

3.7.37 VirtioPCIBlockDevice

virtio block device with PCI transport. This model is written in C++.

VirtioPCIBlockDevice contains the following CADI targets:

- VirtioPCIBlockDevice

VirtioPCIBlockDevice contains the following MTI components:

- *4.106 PVBusMaster - trace* on page 4-2759
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.121 VirtioPCIBlockDevice - trace* on page 4-2872

Table 3-296 Ports

Name	Protocol	Type	Description
info_s	*Value_64* on page 2-85	Slave	Communicates PCI Info values to device.
intr	*Signal* on page 2-85	Master	Virtio device sets interrupt to signal completion.
msixintr[2]	*Signal* on page 2-85	Master	Interrupts for MSI-X table entries.
pvbus	PVBus	Slave	Virtio pci/control/config/status registers.
virtio_m	PVBus	Master	Virtio device performs DMA accesses via master.

3.7.38 VirtualEthernetCrossover

Ethernet Crossover Cable. This model is written in LISA+.

VirtualEthernetCrossover contains the following CADI targets:

- VirtualEthernetCrossover

VirtualEthernetCrossover contains the following MTI components:

- *4.122 VirtualEthernetCrossover - trace* on page 4-2875

About VirtualEthernetCrossover

This component implements two VirtualEthernet slave ports and enables you to connect two VirtualEthernet master ports. It forwards data received on one port to the other port without delay.

Table 3-297 Ports

Name	Protocol	Type	Description
deva	*VirtualEthernet* on page 2-79	Slave	Slave port for connecting to a VirtualEthernet master.
devb	*VirtualEthernet* on page 2-79	Slave	Slave port for connecting to a VirtualEthernet master.

3.7.39 VirtualEthernetHub3

3 Port Ethernet Hub. This model is written in LISA+.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VirtualEthernetHub3 contains the following CADI targets:

- VirtualEthernetHub3

Table 3-298 Ports

Name	Protocol	Type	Description
deva	*VirtualEthernet on page 2-79*	Slave	Slave port for connecting to a VirtualEthernet master.
devb	*VirtualEthernet on page 2-79*	Slave	Slave port for connecting to a VirtualEthernet master.
devc	*VirtualEthernet on page 2-79*	Slave	Slave port for connecting to a VirtualEthernet master.

3.7.40 VisEventRecorder

Event recorder component for visualisation component (allows to playback and record interactive GUI sessions). This model is written in LISA+.

VisEventRecorder contains the following CADI targets:

- ClockDivider
- VisEventRecorder

VisEventRecorder contains the following MTI components:

- *4.60 ClockDivider - trace* on page 4-2438

Table 3-299 Ports

Name	Protocol	Type	Description
control	VisEventRecorderProtocol	Slave	The visualisation component controls the recorder through this port.
ticks	*InstructionCount on page 2-84*	Slave	Allow VisEventRecorder to get tick count from a core.

Table 3-300 Parameters for VisEventRecorder

Name	Type	Default value	Description
checkInstructionCount	bool	0x1	check instruction count in recording file against actual instruction count during playback
playbackFileName	string	""	playback filename (empty string disables playback)
recordingFileName	string	""	recording filename (empty string disables recording)
recordingTimeBase	int	0x5f5e100	timebase in 1/s (relative to the master clock (e.g. 100000000 means 10 nanoseconds resolution simulated time for a 1Hz master clock)) to be used for recording (higher values -> higher time resolution, playback time base is always taken from the playback file)
verbose	int	0x0	enable verbose messages (1=normal, 2=even more)

3.7.41 WarningMemory

Memory that prints warnings, and RAZ/WIs or aborts. This model is written in LISA+.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

WarningMemory contains the following CADI targets:

- WarningMemory

WarningMemory contains the following MTI components:

- *4.107 PVBusSlave - trace* on page 4-2760

Table 3-301 Ports

Name	Protocol	Type	Description
pvbus	PVBus	Slave	Bus slave interface.

Table 3-302 Parameters for WarningMemory

Name	Type	Default value	Description
abort_on_reads	bool	0x0	Abort on reads (read 0 if false)
abort_on_writes	bool	0x0	Abort on writes (ignore if false)
read_data	int	0x0	Data to return on reads, if not aborting
warn_on_reads	bool	0x1	Warn on reads
warn_on_writes	bool	0x1	Warn on writes
warning	string	"Invalid access"	Warning string

3.7.42 v8EmbeddedCrossTrigger_Matrix

v8 Embedded Cross Trigger Matrix. This model is written in C++.

About v8EmbeddedCrossTrigger_Matrix

This is a model of a platform level *Cross Trigger Matrix* (CTM) for connection to the *Cross Trigger Interface* (CTI) ports provided on Armv8-A processors in Fast Models. The combination of the CTI and the CTM provides an architectural model of the CoreSight embedded triggering system.

A single instance of the v8EmbeddedCrossTrigger_Matrix component supports up to four clusters, each containing four cores. For example:

```
cluster0.cti[0] => v8ect.cti[0];
cluster0.cti[1] => v8ect.cti[1];
cluster0.cti[2] => v8ect.cti[2];
cluster0.cti[3] => v8ect.cti[3];
...
cluster4.cti[3] => v8ect.cti[12];
cluster4.cti[3] => v8ect.cti[13];
cluster4.cti[3] => v8ect.cti[14];
cluster4.cti[3] => v8ect.cti[15];
```

Table 3-303 Ports

Name	Protocol	Type	Description
cti[32]	*v8EmbeddedCrossTrigger_controlprotocol* on page 2-84	Slave	-

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

3.8 Signals components

This section describes the Signals components.

This section contains the following subsections:

- *3.8.1 AndGate* on page 3-644.
- *3.8.2 LabellerUserSignals* on page 3-644.
- *3.8.3 OrGate* on page 3-645.
- *3.8.4 SGSignalBuffer* on page 3-645.
- *3.8.5 SignalLogger* on page 3-645.
- *3.8.6 Value64Logger* on page 3-646.
- *3.8.7 ValueLogger* on page 3-646.

3.8.1 AndGate

And Gate. This model is written in LISA+.

AndGate contains the following CADI targets:

- AndGate

About AndGate

This component implements a logical AND of two signal input ports to generate a single output signal. For example, you can use it to combine two interrupt signals.

Table 3-304 Ports

Name	Protocol	Type	Description
input[2]	*Signal* on page 2-85	Slave	2 input signals to be AND'ed.
output	*Signal* on page 2-85	Master	AND'ed output signal.

3.8.2 LabellerUserSignals

This model is written in LISA+.

LabellerUserSignals contains the following CADI targets:

- LabellerUserSignals
- PVBusLogger

LabellerUserSignals contains the following MTI components:

- *4.104 PVBusLogger - trace* on page 4-2756
- *4.105 PVBusMapper - trace* on page 4-2758

Table 3-305 Ports

Name	Protocol	Type	Description
pvbus_m	PVBus	Master	Output with modified UserFlags.
pvbus_s	PVBus	Slave	Unmodified input.

Table 3-306 Parameters for PVBusLogger

Name	Type	Default value	Description
trace_debug	bool	0x0	Enable tracing of debug transactions
trace_snoops	bool	0x0	Enable tracing of ACE snoop requests

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-307 Parameters for LabellerUserSignals

Name	Type	Default value	Description
user	int	0x0	User signal to be applied to transactions

3.8.3 OrGate

Or Gate. This model is written in LISA+.

OrGate contains the following CADI targets:

- OrGate

About OrGate

This component implements a logical OR of two signal input ports to generate a single output signal. For example, you can use this component to combine two interrupt signals.

Table 3-308 Ports

Name	Protocol	Type	Description
input[16]	*Signal* on page 2-85	Slave	16 input signals to be OR'ed.
output	*Signal* on page 2-85	Master	OR'ed output signal.

3.8.4 SGSignalBuffer

Buffer to synchronise SystemGenerator Signal setValue() calls. This model is written in LISA+.

SGSignalBuffer contains the following CADI targets:

- SGSignalBuffer

Note

Variants of this component also exist with multiple input and output ports.

Table 3-309 Ports

Name	Protocol	Type	Description
clk_in	ClockSignal	Slave	Clock rate to release buffered signals.
in	*Signal* on page 2-85	Slave	Signal in.
out	*Signal* on page 2-85	Master	Buffered signal out.

3.8.5 SignalLogger

Traces signal activity. This model is written in LISA+.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SignalLogger contains the following CADI targets:

- SignalLogger

SignalLogger contains the following MTI components:

- *4.113 SignalLogger - trace* on page 4-2833

Table 3-310 Ports

Name	Protocol	Type	Description
in	*Signal* on page 2-85	Slave	Input signal port.
out	*Signal* on page 2-85	Master	Output signal port.

3.8.6 Value64Logger

Traces value activity. This model is written in LISA+.

Value64Logger contains the following CADI targets:

- Value64Logger

Value64Logger contains the following MTI components:

- *4.117 Value64Logger - trace* on page 4-2866

Table 3-311 Ports

Name	Protocol	Type	Description
in	*Value_64* on page 2-85	Slave	Input signal port.
out	*Value_64* on page 2-85	Master	Output signal port.

3.8.7 ValueLogger

Traces value activity. This model is written in LISA+.

ValueLogger contains the following CADI targets:

- ValueLogger

ValueLogger contains the following MTI components:

- *4.118 ValueLogger - trace* on page 4-2867

Table 3-312 Ports

Name	Protocol	Type	Description
in	*Value* on page 2-85	Slave	Input signal port.
out	*Value* on page 2-85	Master	Output signal port.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

3.9 SystemIP components

This section describes the SystemIP components.

The major SystemIP components are:

- Input/output devices.
- Memory, including flash.
- Ethernet controller.
- Interrupt controllers.
- Static and dynamic memory controllers.
- Audio interface.
- Programmable clock generators.

These components are software implementations of specific hardware functionality.

This section contains the following subsections:

- *3.9.1 AHCI_SATA* on page 3-648.
- *3.9.2 BP141_TZMA* on page 3-648.
- *3.9.3 BP147_TZPC* on page 3-650.
- *3.9.4 CCI400* on page 3-650.
- *3.9.5 CCI500* on page 3-652.
- *3.9.6 CCI550* on page 3-656.
- *3.9.7 CCN502* on page 3-660.
- *3.9.8 CCN504* on page 3-662.
- *3.9.9 CCN508* on page 3-664.
- *3.9.10 CCN512* on page 3-665.
- *3.9.11 CMN600* on page 3-667.
- *3.9.12 CMN600CMLHub* on page 3-672.
- *3.9.13 CMN650* on page 3-674.
- *3.9.14 DMC500* on page 3-679.
- *3.9.15 DMC520* on page 3-680.
- *3.9.16 DMC620* on page 3-681.
- *3.9.17 DMC_400* on page 3-682.
- *3.9.18 EthosU55* on page 3-683.
- *3.9.19 GIC500* on page 3-683.
- *3.9.20 GIC500_ClusterPorts* on page 3-688.
- *3.9.21 GIC500_Filter* on page 3-693.
- *3.9.22 GIC600* on page 3-699.
- *3.9.23 GIC600AE* on page 3-702.
- *3.9.24 GIC600AE_Filter* on page 3-712.
- *3.9.25 GIC600_Filter* on page 3-722.
- *3.9.26 GIC_400* on page 3-725.
- *3.9.27 GIC_IRI* on page 3-728.
- *3.9.28 GIC_IRI_Filter* on page 3-737.
- *3.9.29 GICv3IRI* on page 3-746.
- *3.9.30 GICv3IRI_Filter* on page 3-756.
- *3.9.31 ICS307* on page 3-765.
- *3.9.32 IDAU* on page 3-766.
- *3.9.33 MMC* on page 3-767.
- *3.9.34 MMU_400* on page 3-770.
- *3.9.35 MMU_400_BASE* on page 3-777.
- *3.9.36 MMU_500* on page 3-779.
- *3.9.37 MMU_500_BASE* on page 3-786.
- *3.9.38 MMU_600* on page 3-788.
- *3.9.39 MMU_700* on page 3-798.
- *3.9.40 MemoryMappedCounterModule* on page 3-809.
- *3.9.41 MessageHandlingUnitV2* on page 3-810.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

- *3.9.42 PL011_Uart* on page 3-811.
- *3.9.43 PL022_SSP* on page 3-812.
- *3.9.44 PL030_RTC* on page 3-813.
- *3.9.45 PL031_RTC* on page 3-814.
- *3.9.46 PL041_AACI* on page 3-814.
- *3.9.47 PL050_KMI* on page 3-815.
- *3.9.48 PL061_GPIO* on page 3-816.
- *3.9.49 PL080_DMAC* on page 3-816.
- *3.9.50 PL110_CLCD* on page 3-817.
- *3.9.51 PL111_CLCD* on page 3-818.
- *3.9.52 PL180_MCI* on page 3-819.
- *3.9.53 PL192_VIC* on page 3-820.
- *3.9.54 PL310_L2CC* on page 3-821.
- *3.9.55 PL330_DMAC* on page 3-824.
- *3.9.56 PL350_SMC* on page 3-826.
- *3.9.57 PL350_SMC_NAND_FLASH* on page 3-829.
- *3.9.58 PL370_HDLCD* on page 3-830.
- *3.9.59 PL390_GIC* on page 3-830.
- *3.9.60 PPUMTWakerequest* on page 3-833.
- *3.9.61 PPUv0* on page 3-833.
- *3.9.62 PPUv1* on page 3-834.
- *3.9.63 PPUv1_Cluster_Wakerequest_Logic* on page 3-836.
- *3.9.64 SMMUv3AEM* on page 3-836.
- *3.9.65 SMSC_91C111* on page 3-854.
- *3.9.66 SP804_Timer* on page 3-855.
- *3.9.67 SP805_Watchdog* on page 3-855.
- *3.9.68 SP810_SysCtrl* on page 3-855.
- *3.9.69 TZC_400* on page 3-856.
- *3.9.70 TZFilterUnit* on page 3-858.
- *3.9.71 TZIC* on page 3-858.
- *3.9.72 v7_VGIC* on page 3-859.

3.9.1 AHCI_SATA

AHCI controller with attached SATA disks and PCIe interface. This model is written in C++.

AHCI_SATA contains the following CADI targets:

- AHCI_SATA

AHCI_SATA contains the following MTI components:

- *4.1 AHCI_SATA - trace* on page 4-864
- *4.106 PVBusMaster - trace* on page 4-2759
- *4.107 PVBusSlave - trace* on page 4-2760

Table 3-313 Ports

Name	Protocol	Type	Description
ahci_dma_m	PVBus	Master	-
client_s	PCIDevice2ClientProtocol	Slave	-
pvbus	PVBus	Slave	-

3.9.2 BP141_TZMA

PrimeCell Infrastructure AMBA 3 AXI TrustZone Memory Adapter. This model is written in LISA+.

BP141_TZMA contains the following CADI targets:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

- BP141_TZMA
- PVBusRange
- TZSwitch

BP141_TZMA contains the following MTI components:

- *4.105 PVBusMapper - trace* on page 4-2758

About BP141_TZMA

BP141_TZMA permits a single physical memory cell of up to 2MB to be shared between a secure and non-secure storage area. The partitioning between these areas is flexible.

This component routes transactions according to the following:

- The memory region that they are attempting to access.
- The security mode of the transaction.

The BP141_TZMA fixes the base address of the secure region to the base address of the decode space. It uses the R0SIZE[9:0] input to configure the size of the secure region in 4KB increments up to a maximum of 2MB.

`TZMEMSIZE` is the maximum addressing range of the memory as defined by that parameter. By default, `TZMEMSIZE` is set to 2MB. In the following table, AxADDR is the offset address that the transactions want to access.

Table 3-314 BP141_TZMA security control

AxADDR	Memory Region	Non-secure Transfer	Secure Transfer
AxADDR < R0Size	Secure, R0	Illegal	Legal
R0SIZE <= AxADDR and AxADDR < `TZMEMSIZE`	Non-secure, R1	Legal	Legal
AxADDR => `TZMEMSIZE`	No access	Illegal	Illegal

Table 3-315 Ports

Name	Protocol	Type	Description
`R0Size`	*`Value` on page 2-85*	Slave	A software interface that is driven from the TrustZone Protection Controller (TZPC), setting the secure region size by bits[9:0].
`pv_output`	`PVBus`	Master	Routed PVBus output
`pvbus`	`PVBus`	Slave	Bus slave interface.

Table 3-316 Parameters for PVBusRange

Name	Type	Default value	Description
`pvbusrange_0.range`	`int`	`0x0`	Addressable range routed to pvbus_port_a

Table 3-317 Parameters for TZSwitch

Name	Type	Default value	Description
`normal`	`int`	`0x2`	Normal Port
`secure`	`int`	`0x1`	Secure Port

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-318 Parameters for BP141_TZMA

Name	Type	Default value	Description
TZMEMSIZE	int	0x200000	Addressable range of device
TZSECROMSIZE	int	0x200	Default secure size
TZSEGSIZE	int	0x1000	Segment size

3.9.3 BP147_TZPC

TrustZone Protection Controller. This model is written in LISA+.

BP147_TZPC contains the following CADI targets:

- BP147_TZPC

BP147_TZPC contains the following MTI components:

- *4.107 PVBusSlave - trace* on page 4-2760

About BP147_TZPC

BP147_TZPC provides a software interface to the protection bits in a secure system in a TrustZone design.

Table 3-319 Ports

Name	Protocol	Type	Description
TZPCDECPROT0	*Value* on page 2-85	Master	Output decode protection 0 status.
TZPCDECPROT1	*Value* on page 2-85	Master	Output decode protection 1 status.
TZPCDECPROT2	*Value* on page 2-85	Master	Output decode protection 2 status.
TZPCR0SIZE	*Value* on page 2-85	Master	Output secure RAM region size.
bus_in_s	PVBus	Slave	Slave port for register access.

3.9.4 CCI400

Cache Coherent Interconnect for AXI4 ACE. This model is written in LISA+.

CCI400 contains the following CADI targets:

- CCI400

CCI400 contains the following MTI components:

- *4.51 CCI400 - trace* on page 4-2405

About CCI400

- If you disable the `cache_state_modelled` parameter, this component has negligible performance impact. If you enable `cache_state_modelled`, it adds significant cost to throughput for coherent transactions.
- This model implements the slave interface Shareable Override Register, which can be read and written, but it has no functionality.

ACE limitation

AXI Coherency Extensions (ACE) are extensions to AXI4 that support system-level cache-coherency between multiple clusters. The ACE cache models in the Cortex-A15 and the Cortex-A7, and the ACE support in the CCI-400 have the limitation that they only process one transaction at a time. Normally, the simulation processes each transaction to completion before allowing any master to generate another transaction. However, in the following situation the simulation might fail. If a SystemC bus slave calls

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

`wait()` while it is processing a transaction, this call might allow another master to issue another transaction that passes through the CCI-400 or the Cortex-A15/Cortex-A7 caches. This situation could happen if a SystemC bus master running in another thread is connected to one of the ACE-lite ports on the CCI-400.

Table 3-320 Ports

Name	Protocol	Type	Description
acchannelen	*Value* on page 2-85	Slave	For each upstream port, determine if it is enabled or not with respect to snoop requests.
barrierterminate	*Value* on page 2-85	Slave	For each downstream port, determine if barriers are terminated at that port.
broadcastcachemain	*Value* on page 2-85	Slave	For each downstream port, determine if broadcast cache maintenance operations are forwarded down that port. A three bit signal but as the model only have a single downstream port, setting any of the bits will make it work.
bufferableoverride	*Value* on page 2-85	Slave	For each downstream port, determine if all transactions are forced to non-bufferable (AWCACHE[0] is forced to 0).
clk_in	ClockSignal	Slave	Clock signal for cciregisters
errorirq	*Signal* on page 2-85	Master	A signal stating that the imprecise error register is nonzero.
evntcntoverflow[5]	*Signal* on page 2-85	Master	When an event counter overflows, it sets the corresponding signal.
lint_ace_3_reset_state	*Signal* on page 2-85	Slave	This port can be connected to the reset signals of the system attached to the pvbus_s_ace_3 port.
lint_ace_4_reset_state	*Signal* on page 2-85	Slave	This port can be connected to the reset signals of the system attached to the pvbus_s_ace_4 port.
periphbase	*Value_64* on page 2-85	Slave	This port sets the base address of the private peripheral region.
pvbus_m	PVBus	Master	Master port for all downstream memory accesses.
pvbus_s_ace_3	PVBus	Slave	ACE-capable slave ports.
pvbus_s_ace_4	PVBus	Slave	ACE-capable slave ports.
pvbus_s_ace_lite_plus_dvm_0	PVBus	Slave	Memory bus interface that implements ACE lite and DVM protocol.
pvbus_s_ace_lite_plus_dvm_1	PVBus	Slave	Memory bus interface that implements ACE lite and DVM protocol.
pvbus_s_ace_lite_plus_dvm_2	PVBus	Slave	Memory bus interface that implements ACE lite and DVM protocol.
reset_in	*Signal* on page 2-85	Slave	Signal to reset the CCI.
reset_state_of_ace_lite_ports[3]	*Signal* on page 2-85	Slave	This port can be connected to the reset signals of the system attached to ACE-Lite ports 0,1,2

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-321 Parameters for CCI400

Name	Type	Default value	Description
acchannelen	int	0x1f	For each upstream port, determine if it is enabled or not w.r.t. snoop requests.
barrierterminate	int	0x7	For each downstream port, determine if barriers will be terminated at that port.
broadcastcachemain	int	0x0	For each downstream port a bit determines if broadcast cache maintenance operations are forwarded down that port.
bufferableoverride	int	0x0	For each downstream port, determine if all transactions will be forced to non-bufferable.
cache_state_modelled	bool	0x1	Model the cache state.
force_on_from_start	bool	0x0	The CCI will normally start up with snooping disabled, however, using this then we allow the model to start up as enabled without having to program it. This is only setup at simulation reset and not at signal reset. If the upstreams can ever be held in reset then you *must* connect the reset_state_of_ace_lite_ports[], lint_ace_3_reset_state and lint_ace_4_reset_state so that it knows when to disable snoops to the upstream systems. Otherwise, the upstream system will receive snoop messages whilst in reset and it will complain that it 'received a snoop request whilst it was in reset'.
log_enabled	int	0x1	Enable log messages from the CCI register file. Log level 0 means do not print anything, 1 means print only access violations, 2 means also print writes, 3 means print reads as well.
periphbase	int	0x2c000000	Value for PERIPHBASE. Only bits [39:16] are used. This value may be overriden by an input on the periphbase port
revision	string	"r0p0"	Revision of the CCI400

3.9.5 CCI500

Cache Coherent Interconnect. This model is written in C++.

CCI500 contains the following CADI targets:

- CCI500

CCI500 contains the following MTI components:

- *4.52 CCI500 - trace* on page 4-2407
- *4.107 PVBusSlave - trace* on page 4-2760

About CCI500

The LISA file declares seven upstream ports. You can configure these ports with `num_ace_ports` and `num_ace_lite_ports`. The bottom `num_ace_lite_ports` are ACE-Lite+DVM. The next `num_ace_ports` are ACE. Any remaining ports are ignored. If transactions are made on them, then warnings are produced. For example, if `num_ace_ports` = 1 and `num_ace_lite_ports` = 1 then `pvbus_s[1]` is ACE, `pvbus_s[0]` is ACE-Lite+DVM and `pvbus_s[6-2]` are considered not to exist.

Differences between the model and the RTL

Address Decoder

- Only supports striping down to 4KiB.
- If the address decoder aborts the access, returns `SLVERR` rather than `DECERR`.

Interfaces

The model does not implement the Q-Channel and P-Channel interfaces.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Performance Monitoring Unit

- PMU counters recognize only a few event sources:
  - Slave interface events:
    - **3** ReadOnce.
    - **4** ReadClean, ReadShared, ReadNotSharedDirty, ReadUnique.
    - **5** MakeUnique, CleanUnique.
    - **6** CleanInvalid, CleanShared, MakeInvalid.
    - **7** DVM transaction received from upstream.
    - **9** Read data that is satisfied by a snoop request.
  - No events are implemented for the global events or for the master events.
- The PMU does not implement the event bus (`EVNTBUS`).

Register access

The register file only supports 32-bit accesses to its registers. Later versions of the CCI500 hardware support full write strobes to the register file. This limitation means that byte and halfword accesses work on the hardware but not on this version of the component.

Snoop filter RAMs

Snoop filter RAMs are not modeled. The Status Register fields that relate to the power state of the Snoop filter RAMs are undefined.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-322 Ports

Name	Protocol	Type	Description
acchannelensx[7]	*Value* on page 2-85	Slave	ACCHANNELENSx represents the ports ACCHANNELENS0..ACCHANNELENS7 on the RTL (assuming there are seven upstream ports). * each upstream ACE port 'y' (pvbus_s[y]) has a two bit ACCHANNELENSx * bit 0 == 0 -- DVM messages are disabled from being sent to this interface * bit 1 == 0 -- Snoop messages are disabled from being sent to this interface * each upstream ACE-Lite port 'z' (pvbus_s[z]) has a one bit ACCHANNELENSx * bit 0 == 0 -- DVM messages are disabled from being sent to this interface In the model, as we support a variety of configurations with a single LISA file then each port will behave as though it is one bit or two bit as appropriate. If you send a value that cannot be represented, given the width of the port, then the CCI model will halt and produce a fatal error. The assumed values of these are set by parameters until they are driven, so you need not drive them if they are constant. In the RTL, these signals are sampled at reset. Due to ordering issues w.r.t. reset() on different components then we cannot do that. Instead the signals are sampled at first transaction. Thus any controller that is producing these signals has to hold them constant for long enough. AC channel enables.
address_decoder	CCI500_AddressDecoderProtocol	Master	An address decoder can be attached to the address_decoder port to choose which pvbus_s port a downstream transaction will go out of. If you do not connect an address decoder then all transactions will go out of port 0.
dbgen	*Signal* on page 2-85	Slave	Invasive debug enable.
errirq	*Signal* on page 2-85	Master	Indicates that an error response, DECERR or SLVERR, is received on the RRESP, BRESP, or CRRESP input signals, and it cannot be signaled precisely.
evntcntoverflow[8]	*Signal* on page 2-85	Master	Overflow flags for the PMU clock and counters.
niden	*Signal* on page 2-85	Slave	Non-invasive debug enable.
pvbus_m[6]	PVBus	Master	Bus master ports.
pvbus_register_file_s	PVBus	Slave	The slave port of the register file.
pvbus_s[7]	PVBus	Slave	Bus slave ports.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-322 Ports (continued)

Name	Protocol	Type	Description
reset_in	*Signal* on page 2-85	Slave	Reset the interconnect.
reset_state_of_upstream_port[7]	*Signal* on page 2-85	Slave	Tell the interconnect the reset state of the upstream ports, this can be used by the interconnect to check some aspects of the reset sequencing. If you are using force_on_from_start then you _must_ connect these pins.
spiden	*Signal* on page 2-85	Slave	Secure invasive debug enable.
spniden	*Signal* on page 2-85	Slave	Secure privileged non-invasive debug enable.

Table 3-323 Parameters for CCI500

Name	Type	Default value	Description
acchannelens0	int	0x0	For upstream port 0 determine if it is enabled or not w.r.t. snoop requests.
acchannelens1	int	0x0	For upstream port 1 determine if it is enabled or not w.r.t. snoop requests.
acchannelens2	int	0x0	For upstream port 2 determine if it is enabled or not w.r.t. snoop requests.
acchannelens3	int	0x0	For upstream port 3 determine if it is enabled or not w.r.t. snoop requests.
acchannelens4	int	0x0	For upstream port 4 determine if it is enabled or not w.r.t. snoop requests.
acchannelens5	int	0x0	For upstream port 5 determine if it is enabled or not w.r.t. snoop requests.
acchannelens6	int	0x0	For upstream port 6 determine if it is enabled or not w.r.t. snoop requests.
addr_width	int	0x28	The bit-width of the address that the CCI can accept.
cache_state_modelled	bool	0x1	Model the cache state.
dbgen	bool	0x1	Invasive debug enable. If true, enables the counting of PMU events.
enable_logger	bool	0x0	Enable PVBusLoggers for the downstream ports in the CCI model.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-323 Parameters for CCI500 (continued)

Name	Type	Default value	Description
force_on_from_start	bool	0x0	The interconnect will normally start up with snooping/DVM disabled. This parameter enables the upstream system of a port to be snooped if the upstream is not in reset and if ACCHANNELENSx allows it. No software driver for the interconnect is needed. Any port that could go into reset must have 'reset_state_of_upstream_port[]' reflect the reset state of that upstream system. Otherwise, the upstream system may receive snoop/DVM messages whilst in reset and may complain that it 'received a snoop request whilst it was in reset'. Do not use if software is directly controlling the interconnect. This option does not disavow the responsibility of the upstream system to clean any shared dirty data from its caches before going into reset.
niden	bool	0x1	Whether non-secure events are allowed to be counted in the performance monitor
num_ace_lite_ports	int	0x5	The bottom num_ace_lite_ports are ACE-Lite+DVM.
num_ace_ports	int	0x2	The top num_ace_ports are ACE and support full coherency.
number_of_phantom_entries	int	0x20	Number of phantom entries in the cache. Phantom entries are used by certain cache operations to hold temporary data. Usually this should be left at the default value which is safe for all systems containing up to 32 masters.
qos_threshold_upper	int	0xc	Reset value for the QoS threshold register.
reentrancy_support	string	"env"	Must be one of: on/off/cacheglobal/env: 'on': hazard checking per cache line (normal mode), 'off': no hazard checking (use only for single master systems), 'cacheglobal': hazard checking globally for cache (not per cache line, testing feature, provokes more hazards than necessary), 'env' (or empty string): take value from FM_REENTRANCY_SUPPORT env var, if this is not set use 'on', default is 'env'
spiden	bool	0x1	Secure invasive debug enable. If both SPIDEN and DBGEN are high, enables the counting of both Non-secure and Secure events.
spniden	bool	0x1	Whether secure and non-secure events are allowed to be counted in the performance monitor
version	string	""	The version of the interconnect. Allowed versions are:- r0p0, r0p2, r1p0

3.9.6 CCI550

Cache Coherent Interconnect for AXI4 ACE. This model is written in C++.

CCI550 contains the following CADI targets:

- CCI550

CCI550 contains the following MTI components:

- *4.53 CCI550 - trace* on page 4-2412
- *4.107 PVBusSlave - trace* on page 4-2760

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Differences between the model and the RTL

Address Decoder

- Only supports striping down to 4KiB.
- If the address decoder aborts the access, returns `SLVERR` rather than `DECERR`.

Interfaces

The model does not implement the Q-Channel and P-Channel interfaces.

Performance Monitoring Unit

- PMU counters recognize only a few event sources:
  - Slave interface events:
    - **3** ReadOnce.
    - **4** ReadClean, ReadShared, ReadNotSharedDirty, ReadUnique.
    - **5** MakeUnique, CleanUnique.
    - **6** CleanInvalid, CleanShared, MakeInvalid.
    - **7** DVM transaction received from upstream.
    - **9** Read data that is satisfied by a snoop request.
  - No events are implemented for the global events or for the master events.
- The PMU does not implement the event bus (`EVNTBUS`).

Reset signal sampling

The configuration ports `acchannelensx[]` are sampled in the hardware when coming out of reset. In the model, these ports are sampled at the first transaction to a `pvbus_s` port or to the register file.

Status Register, change-pending, and DVM messages

The Status Register provides information on when the last transaction that could have observed an old value of a snoop or DVM enable has finished in the upstream system. Therefore a port that has been disabled can now have the system upstream of that port turned off. The model does not track DVM messages in the upstream system.

Snoop filter RAMs

- The CCI-550 hardware has a snoop filter that reduces the number of snoop requests that the interconnect has to make. The model does not have a snoop filter and could make more snoop requests than the hardware would. This difference has no programmer-visible effect.
- The Status Register fields that relate to the power state of the Snoop filter RAMs are undefined.

Registers

The following registers provide storage but have no effect on the model.

- QoS registers.
- Interface monitor registers. These registers are intended for silicon debug.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-324 Ports

Name	Protocol	Type	Description
acchannelensx[7]	*Value* on page 2-85	Slave	The acchannelensx[N] pins are used to tell the interconnect if the upstream system will accept snoops and/or DVM messages.
address_decoder	CCI500_AddressDecoderProtocol	Master	An address decoder can be attached to the address_decoder port to choose which pvbus_s port a downstream transaction will go out of. If you do not connect an address decoder then all transactions will go out of port 0.
dbgen	*Signal* on page 2-85	Slave	Invasive debug enable.
errirq	*Signal* on page 2-85	Master	Some async error was detected.
evntcntoverflow[8]	*Signal* on page 2-85	Master	The output interrupts of the event counters.
niden	*Signal* on page 2-85	Slave	Non-invasive debug enable.
pvbus_m[7]	PVBus	Master	The downstream master ports.
pvbus_register_file_s	PVBus	Slave	The slave port of the register file.
pvbus_s[7]	PVBus	Slave	Bus slave ports.
reset_in	*Signal* on page 2-85	Slave	Reset the interconnect.
reset_state_of_upstream_port[7]	*Signal* on page 2-85	Slave	Tell the interconnect the reset state of the upstream ports, this can be used by the interconnect to check some aspects of the reset sequencing. If you are using force_on_from_start then you _must_ connect these pins.
sci_s[7]	SystemCoherencyInterface	Slave	The System Coherency Interface bus. For those upstream ports that have a corresponding bit set in the bitmap of si_system_coherency_interface then the corresponding sci_m port can be used to move the upstream system into and out of the coherency domain.
spiden	*Signal* on page 2-85	Slave	Secure privileged invasive debug enable.
spniden	*Signal* on page 2-85	Slave	Secure privileged non-invasive debug enable.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-325 Parameters for CCI550

Name	Type	Default value	Description
acchannelens0	int	0x0	For upstream port 0 determine if it is enabled or not w.r.t. snoop requests.
acchannelens1	int	0x0	For upstream port 1 determine if it is enabled or not w.r.t. snoop requests.
acchannelens2	int	0x0	For upstream port 2 determine if it is enabled or not w.r.t. snoop requests.
acchannelens3	int	0x0	For upstream port 3 determine if it is enabled or not w.r.t. snoop requests.
acchannelens4	int	0x0	For upstream port 4 determine if it is enabled or not w.r.t. snoop requests.
acchannelens5	int	0x0	For upstream port 5 determine if it is enabled or not w.r.t. snoop requests.
acchannelens6	int	0x0	For upstream port 6 determine if it is enabled or not w.r.t. snoop requests.
addr_width	int	0x28	The bit-width of the address that the CCI can accept.
cache_state_modelled	bool	0x1	Model the cache state.
dbgen	bool	0x1	Invasive debug enable. If true, enables the counting of PMU events.
enable_logger	bool	0x0	Enable PVBusLoggers for the downstream ports in the CCI model.
force_on_from_start	bool	0x0	The interconnect will normally start up with snooping/DVM disabled. The parameter si_system_coherency_interface determines which connections are managed by the System Coherency Interface (SCI). For connections that are managed by SCI, then this parameter has no effect. For all other connections, this parameter enables the upstream system of a port to be snooped if the upstream is not in reset and if ACCHANNELENSx allows it. No software driver for the interconnect is needed. Any non-SCI port that could go into reset must have 'reset_state_of_upstream_port[]' reflect the reset state of that upstream system. Otherwise, the upstream system may receive snoop/DVM messages whilst in reset and may complain that it 'received a snoop request whilst it was in reset'. Do not use if software is directly controlling the interconnect. This option does not disavow the responsibility of the upstream system to clean any shared dirty data from its caches before going into reset.
niden	bool	0x1	Whether non-secure events are allowed to be counted in the performance monitor
num_ace_lite_ports	int	0x5	The bottom num_ace_lite_ports are ACE-Lite+DVM.
num_ace_ports	int	0x2	The top num_ace_ports are ACE and support full coherency.
number_of_phantom_entries	int	0x20	Number of phantom entries in the cache. Phantom entries are used by certain cache operations to hold temporary data. Usually this should be left at the default value which is safe for all systems containing up to 32 masters.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-325 Parameters for CCI550 (continued)

Name	Type	Default value	Description
qos_threshold_upper	int	0xc	Reset value for the QoS threshold register.
reentrancy_support	string	"env"	Must be one of: on/off/cacheglobal/env: 'on': hazard checking per cache line (normal mode), 'off': no hazard checking (use only for single master systems), 'cacheglobal': hazard checking globally for cache (not per cache line, testing feature, provokes more hazards than necessary), 'env' (or empty string): take value from FM_REENTRANCY_SUPPORT env var, if this is not set use 'on', default is 'env'
si0_qos_bw_regulator	bool	0x0	For upstream port 0 determine if it has a BW regulator. The effect of QoS is not modelled and this parameter only alters some registers.
si1_qos_bw_regulator	bool	0x0	For upstream port 1 determine if it has a BW regulator. The effect of QoS is not modelled and this parameter only alters some registers.
si2_qos_bw_regulator	bool	0x0	For upstream port 2 determine if it has a BW regulator. The effect of QoS is not modelled and this parameter only alters some registers.
si3_qos_bw_regulator	bool	0x0	For upstream port 3 determine if it has a BW regulator. The effect of QoS is not modelled and this parameter only alters some registers.
si4_qos_bw_regulator	bool	0x0	For upstream port 4 determine if it has a BW regulator. The effect of QoS is not modelled and this parameter only alters some registers.
si5_qos_bw_regulator	bool	0x0	For upstream port 5 determine if it has a BW regulator. The effect of QoS is not modelled and this parameter only alters some registers.
si6_qos_bw_regulator	bool	0x0	For upstream port 6 determine if it has a BW regulator. The effect of QoS is not modelled and this parameter only alters some registers.
si_system_coherency_interface	int	0x0	This parameter tells the interconnect which upstream ports should be controlled by the System Coherency Interface. Each bit corresponds to an upstream port, bit 0 to upstream port 0, etc. If the SCI port is connected but si_system_coherency_interface disable its use then messages from the upstream will be ignored and software must manage the upstream system's entrance and exit of the coherency domain.
spiden	bool	0x1	Secure invasive debug enable. If both SPIDEN and DBGEN are high, enables the counting of both Non-secure and Secure events.
spniden	bool	0x1	Whether secure and non-secure events are allowed to be counted in the performance monitor
version	string	""	The version of the interconnect. Allowed versions are:- r0p0, r1p0

3.9.7 CCN502

CCN502 Interconnect Fast Model. This model is written in C++.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CCN502 contains the following CADI targets:

- CCN5XX
- CCNCache

CCN502 contains the following MTI components:

- *4.54 CCNCache - trace* on page 4-2417
- *4.55 CCNRegisterSet - trace* on page 4-2422
- *4.102 PVBusExclusiveMonitor - trace* on page 4-2752
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760

About CCN502

CCN502 can be used for connecting components using the ACE and ACE-Lite interfaces. It has an L3 cache that can provide coherency between up to four fully coherent ACE clusters and nine I/O coherent masters. It can connect up to two memory elements to drive transaction requests. This interconnect can be configured to support six or eight crosspoints, both of which are implemented in the model.

CCN502 has three or five downstream ports, depending on the number of crosspoints:

- Two or four SNF ports for the memory controller
- One Acelite port (HNI)

Some useful parameters are:

- `cache_state_modelled`
- `cache_size_in_kbytes`
- `systemaddrmap`

The model has the following limitations:

- No support for 3 SN striping.
- If there are multiple SN-Fs, the distribution of addresses to each SN-F is not guaranteed to match the hardware.
- The parameters for the `CCNCache` subcomponent are not accessible in System Canvas.

Table 3-326 Ports

Name	Protocol	Type	Description
`pvbus_m_hni[1]`	`PVBus`	Master	HNI downstream port.
`pvbus_m_snf[4]`	`PVBus`	Master	SNF downstream ports.
`pvbus_s_rnf[4]`	`PVBus`	Slave	RNF upstream ports.
`pvbus_s_rni[9]`	`PVBus`	Slave	RNI upstream ports
`reset_in`	*Signal* on page 2-85	Slave	Reset signal.

Table 3-327 Parameters for CCNCache

Name	Type	Default value	Description
`ccn_cache.acchannelen_rnf`	`int`	`0xf`	For each upstream port, determine if it is enabled or not w.r.t. snoop requests.
`ccn_cache.acchannelen_rni`	`int`	`0xfff`	For each upstream port, determine if it is enabled or not w.r.t. dvm requests.
`ccn_cache.cache_size_kb`	`int`	`0x2000`	Number of kilo bytes in cache
`ccn_cache.cache_state_modelled`	`bool`	`0x1`	Model the cache state.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-327 Parameters for CCNCache (continued)

Name	Type	Default value	Description
ccn_cache.force_on_from_start	bool	0x0	The CCI will normally start up with snooping disabled, however, using this then we allow the model to start up as enabled without having to program it. This is only setup at simulation reset and not at signal reset. If the upstreams can ever be held in reset then you *must* connect the reset_state_of_upstream_port_3 and reset_state_of_upstream_port_4 so that it knows when to disable snoops to the upstream systems. Otherwise, the upstream system will receive snoop messages whilst in reset and it will complain that it 'received a snoop request whilst it was in reset'.
ccn_cache.number_of_phantom_entries	int	0x20	Number of phantom entries in the cache. Phantom entries are used by certain cache operations to hold temporary data. Usually this should be left at the default value which is safe for all systems containing up to 32 masters.
ccn_cache.periphbase	int	0x2c000000	Value for PERIPHBASE. Only bits [43:24] are used
ccn_cache.reentrancy_support	string	"env"	Must be one of: on/off/cacheglobal/env: 'on': hazard checking per cache line (normal mode), 'off': no hazard checking (use only for single master systems), 'cacheglobal': hazard checking globally for cache (not per cache line, testing feature, provokes more hazards than necessary), 'env' (or empty string): take value from FM_REENTRANCY_SUPPORT env var, if this is not set use 'on', default is 'env'
ccn_cache.sbas_bridge_present	bool	0x1	For each upstream port, determine if it is enabled or not w.r.t. dvm requests.
ccn_cache.sbsx_bridge_present	bool	0x1	For each upstream port, determine if it is enabled or not w.r.t. dvm requests.

3.9.8 CCN504

CCN504 Interconnect Fast Model. This model is written in C++.

CCN504 contains the following CADI targets:

- CCN5XX
- CCNCache

CCN504 contains the following MTI components:

- *4.54 CCNCache - trace* on page 4-2417
- *4.55 CCNRegisterSet - trace* on page 4-2422
- *4.102 PVBusExclusiveMonitor - trace* on page 4-2752
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760

About CCN504

CCN504 has three downstream ports:

- Two SNF ports for the memory controller.
- One Acelite port (HNI).

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Note

The parameters for the `CCNCache` subcomponent are not accessible in System Canvas.

Table 3-328 Ports

Name	Protocol	Type	Description
`pvbus_m_hni[1]`	`PVBus`	Master	HNI downstream port.
`pvbus_m_snf[2]`	`PVBus`	Master	SNF downstream ports.
`pvbus_s_rnf[4]`	`PVBus`	Slave	RNF upstream ports.
`pvbus_s_rni[18]`	`PVBus`	Slave	RNI upstream ports.
`reset_in`	*Signal* on page 2-85	Slave	Reset signal.

Table 3-329 Parameters for CCNCache

Name	Type	Default value	Description
`ccn_cache.acchannelen_rnf`	`int`	`0xf`	For each upstream port, determine if it is enabled or not w.r.t. snoop requests.
`ccn_cache.acchannelen_rni`	`int`	`0xfff`	For each upstream port, determine if it is enabled or not w.r.t. dvm requests.
`ccn_cache.cache_size_kb`	`int`	`0x2000`	Number of kilo bytes in cache
`ccn_cache.cache_state_modelled`	`bool`	`0x1`	Model the cache state.
`ccn_cache.force_on_from_start`	`bool`	`0x0`	The CCI will normally start up with snooping disabled, however, using this then we allow the model to start up as enabled without having to program it. This is only setup at simulation reset and not at signal reset. If the upstreams can ever be held in reset then you *must* connect the reset_state_of_upstream_port_3 and reset_state_of_upstream_port_4 so that it knows when to disable snoops to the upstream systems. Otherwise, the upstream system will receive snoop messages whilst in reset and it will complain that it 'received a snoop request whilst it was in reset'.
`ccn_cache.number_of_phantom_entries`	`int`	`0x20`	Number of phantom entries in the cache. Phantom entries are used by certain cache operations to hold temporary data. Usually this should be left at the default value which is safe for all systems containing up to 32 masters.
`ccn_cache.periphbase`	`int`	`0x2c000000`	Value for PERIPHBASE. Only bits [43:24] are used
`ccn_cache.reentrancy_support`	`string`	"env"	Must be one of: on/off/cacheglobal/env: 'on': hazard checking per cache line (normal mode), 'off': no hazard checking (use only for single master systems), 'cacheglobal': hazard checking globally for cache (not per cache line, testing feature, provokes more hazards than necessary), 'env' (or empty string): take value from FM_REENTRANCY_SUPPORT env var, if this is not set use 'on', default is 'env'

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-329 Parameters for CCNCache (continued)

Name	Type	Default value	Description
`ccn_cache.sbas_bridge_present`	`bool`	`0x1`	For each upstream port, determine if it is enabled or not w.r.t. dvm requests.
`ccn_cache.sbsx_bridge_present`	`bool`	`0x1`	For each upstream port, determine if it is enabled or not w.r.t. dvm requests.

3.9.9 CCN508

CCN508 Interconnect Fast Model. This model is written in C++.

CCN508 contains the following CADI targets:

- CCN5XX
- CCNCache

CCN508 contains the following MTI components:

- *4.54 CCNCache - trace* on page 4-2417
- *4.55 CCNRegisterSet - trace* on page 4-2422
- *4.102 PVBusExclusiveMonitor - trace* on page 4-2752
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760

About CCN508

CCN508 has six downstream ports:

- Four SNF ports for the memory controller.
- Two Acelite ports (HNI).

Note

The parameters for the `CCNCache` subcomponent are not accessible in System Canvas.

Table 3-330 Ports

Name	Protocol	Type	Description
`pvbus_m_hni[2]`	`PVBus`	Master	HNI downstream ports.
`pvbus_m_snf[4]`	`PVBus`	Master	SNF downstream ports.
`pvbus_s_rnf[8]`	`PVBus`	Slave	RNF upstream ports.
`pvbus_s_rni[24]`	`PVBus`	Slave	RNI upstream ports.
`reset_in`	*`Signal` on page 2-85*	Slave	Reset signal.

Table 3-331 Parameters for CCNCache

Name	Type	Default value	Description
`ccn_cache.acchannelen_rnf`	`int`	`0xf`	For each upstream port, determine if it is enabled or not w.r.t. snoop requests.
`ccn_cache.acchannelen_rni`	`int`	`0xfff`	For each upstream port, determine if it is enabled or not w.r.t. dvm requests.
`ccn_cache.cache_size_kb`	`int`	`0x2000`	Number of kilo bytes in cache

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-331 Parameters for CCNCache (continued)

Name	Type	Default value	Description
ccn_cache.cache_state_modelled	bool	0x1	Model the cache state.
ccn_cache.force_on_from_start	bool	0x0	The CCI will normally start up with snooping disabled, however, using this then we allow the model to start up as enabled without having to program it. This is only setup at simulation reset and not at signal reset. If the upstreams can ever be held in reset then you *must* connect the reset_state_of_upstream_port_3 and reset_state_of_upstream_port_4 so that it knows when to disable snoops to the upstream systems. Otherwise, the upstream system will receive snoop messages whilst in reset and it will complain that it 'received a snoop request whilst it was in reset'.
ccn_cache.number_of_phantom_entries	int	0x20	Number of phantom entries in the cache. Phantom entries are used by certain cache operations to hold temporary data. Usually this should be left at the default value which is safe for all systems containing up to 32 masters.
ccn_cache.periphbase	int	0x2c000000	Value for PERIPHBASE. Only bits [43:24] are used
ccn_cache.reentrancy_support	string	"env"	Must be one of: on/off/cacheglobal/env: 'on': hazard checking per cache line (normal mode), 'off': no hazard checking (use only for single master systems), 'cacheglobal': hazard checking globally for cache (not per cache line, testing feature, provokes more hazards than necessary), 'env' (or empty string): take value from FM_REENTRANCY_SUPPORT env var, if this is not set use 'on', default is 'env'
ccn_cache.sbas_bridge_present	bool	0x1	For each upstream port, determine if it is enabled or not w.r.t. dvm requests.
ccn_cache.sbsx_bridge_present	bool	0x1	For each upstream port, determine if it is enabled or not w.r.t. dvm requests.

3.9.10 CCN512

CCN512 Interconnect Fast Model. This model is written in C++.

CCN512 contains the following CADI targets:

- CCN5XX
- CCNCache

CCN512 contains the following MTI components:

- *4.54 CCNCache - trace* on page 4-2417
- *4.55 CCNRegisterSet - trace* on page 4-2422
- *4.102 PVBusExclusiveMonitor - trace* on page 4-2752
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760

About CCN512

CCN512 has an L3 cache that can provide coherency between up to 12 fully coherent ACE clusters and 24 I/O coherent masters. It can connect up to four memory elements to drive transaction requests.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CCN512 has six downstream ports:

- Four SNF ports for the memory controller
- Two Acelite ports (HNI)

Some useful parameters are:

- `cache_state_modelled`
- `cache_size_in_mbytes`
- `systemaddrmap`

The model has the following limitations:

- No support for 3 SN striping.
- If there are multiple SN-Fs, the distribution of addresses to each SN-F is not guaranteed to match the hardware.
- The parameters for the `CCNCache` subcomponent are not accessible in System Canvas.

Table 3-332 Ports

Name	Protocol	Type	Description
`pvbus_m_hni[2]`	`PVBus`	Master	HNI downstream ports.
`pvbus_m_snf[4]`	`PVBus`	Master	SNF downstream ports.
`pvbus_s_rnf[12]`	`PVBus`	Slave	RNF upstream ports.
`pvbus_s_rni[24]`	`PVBus`	Slave	RNI upstream ports.
`reset_in`	*Signal* on page 2-85	Slave	Reset signal.

Table 3-333 Parameters for CCNCache

Name	Type	Default value	Description
`ccn_cache.acchannelen_rnf`	`int`	`0xf`	For each upstream port, determine if it is enabled or not w.r.t. snoop requests.
`ccn_cache.acchannelen_rni`	`int`	`0xfff`	For each upstream port, determine if it is enabled or not w.r.t. dvm requests.
`ccn_cache.cache_size_kb`	`int`	`0x2000`	Number of kilo bytes in cache
`ccn_cache.cache_state_modelled`	`bool`	`0x1`	Model the cache state.
`ccn_cache.force_on_from_start`	`bool`	`0x0`	The CCI will normally start up with snooping disabled, however, using this then we allow the model to start up as enabled without having to program it. This is only setup at simulation reset and not at signal reset. If the upstreams can ever be held in reset then you *must* connect the reset_state_of_upstream_port_3 and reset_state_of_upstream_port_4 so that it knows when to disable snoops to the upstream systems. Otherwise, the upstream system will receive snoop messages whilst in reset and it will complain that it 'received a snoop request whilst it was in reset'.
`ccn_cache.number_of_phantom_entries`	`int`	`0x20`	Number of phantom entries in the cache. Phantom entries are used by certain cache operations to hold temporary data. Usually this should be left at the default value which is safe for all systems containing up to 32 masters.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-333 Parameters for CCNCache (continued)

Name	Type	Default value	Description
ccn_cache.periphbase	int	0x2c000000	Value for PERIPHBASE. Only bits [43:24] are used
ccn_cache.reentrancy_support	string	"env"	Must be one of: on/off/cacheglobal/env: 'on': hazard checking per cache line (normal mode), 'off': no hazard checking (use only for single master systems), 'cacheglobal': hazard checking globally for cache (not per cache line, testing feature, provokes more hazards than necessary), 'env' (or empty string): take value from FM_REENTRANCY_SUPPORT env var, if this is not set use 'on', default is 'env'
ccn_cache.sbas_bridge_present	bool	0x1	For each upstream port, determine if it is enabled or not w.r.t. dvm requests.
ccn_cache.sbsx_bridge_present	bool	0x1	For each upstream port, determine if it is enabled or not w.r.t. dvm requests.

3.9.11 CMN600

CMN600 Interconnect Fast Model. This model is written in C++ and models version r1p1 or r3p0 of the RTL.

CMN600 contains the following CADI targets:

- CMN600
- CMN600Cache

CMN600 contains the following MTI components:

- *4.56 CMN600 - trace* on page 4-2424
- *4.58 CMN600Cache - trace* on page 4-2429
- *4.102 PVBusExclusiveMonitor - trace* on page 4-2752
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760

About CMN600

The CMN600 model supports the following:

- rnf, rni/rnd, hni, and snf/sbsx interface ports. The mapping between the port number and `NodeId` is based on the `NodeId` index. For example, RNF2 controls `pvbus_s_rnf[2]`. Its index is specified in the `node_info` register as `logical_id`. Similar behavior can be expected for HNI.

  If both RND and RNI nodes are present, then all starting rei ports are mapped to RND nodes and then the RNI nodes. For example, for CMN600 or CMN650 with two RND nodes, one RNI node, and given each RNI/RND node controls three interface ports, `pvbus_s_rni[0-2]` maps to RND0, `pvbus_s_rni[3-5]` maps to RND1 and `pvbus_s_rni[6-8]` maps to RNI0. Similarly, SNF and SBSX nodes are mapped to `pvbus_m_snf[]` ports where starting ports are mapped to SNF and then SBSX nodes.
- A parameter to define the mesh placement of the CHI nodes. The parameter `mesh_config_file` can be used in the following ways:
  - This way is recommended. Set `mesh_config_file` to the name of the yml configuration file emitted by Socrates™.
  - This way is deprecated. Set `mesh_config_file` to a JSON file, with a `.json` file extension. The JSON file has the following format:

```
{
    "mesh_row_size":4,
    "mesh_column_size":3,
```


Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

```
    "mesh_placement" : {
    "0.0" : "RND:HND",  "1.0" : "RND:RNF",  "2.0" : "RND:RNF",
    "0.1" : "HNF:HNF",  "1.1" : "HNF:HNF",  "2.1" : "HNF:HNF",
    "0.2" : "HNF:HNF",  "1.2" : "HNF:HNF",  "2.2" : "HNF:HNF",
    "0.3" : "RND:HNF",  "1.3" : "HNF:HNF",  "2.3" : "HNF:SNF"
    }
}
```

In the `mesh_placement` tag:

- The value to the left of a colon (`:`) indicates the position of the XP. For example, `"2.3"` means the XP is positioned at coordinates X=2 and Y=3.
- The value to the right of a colon (`:`) indicates the node protocols for port 0 and port 1. For example, `"RND:HND"` means RND is attached to port 0 and HND is attached to port 1.

- The discovery feature to determine the system address of CHI nodes.
- Hashed and non-hashed memory regions, and the `sys_cache_group` for hashed memory regions.
- RN SAM and HN-F SAM memory partition size of 512MB (CMN-600 r0) and 64MB (CMN-600 r1). To select the desired partition size, use the parameter `sam_mem_partition_granularity`. This parameter can be set to either "64MB" or "512MB".
- DVM filtering using VMF registers.
- Entry/exit of RNs from snoop and DVM domains using both SYSCOREQ/SYSCOACK signals and through software using MXP syscoreq registers.
- Reading the content of the cache through HNF l3_cache_access* registers, if `cache_state_modelled` is true.
- HN-F SAM feature to allocate a range of addresses to a specific SNF.
- HN-F Cache way lock and OCM.
- The model implements version r1p1 of the RTL specification and includes CAL (Core Aggregation Layer) functionality added in r3p0.
- The `revision` parameter selects between the r1p1 and r3p0 implementations.
- CAL r2 and r3 features are supported at release quality. Further testing is planned.
- Set the environment variable `FASTSIM_CMN_DUMP_CONFIG` to 1 to print to `stdout` details of the mesh configuration and memory map for the CMN600 model.

  ——— **Note** ———

  `FASTSIM_CMN_DUMP_CONFIG` is deprecated in Fast Models release 11.12. Use the `print_cmn_ccix_config` parameter instead.

  ———————————

Model limitations

- PMU counters are not supported. Counter registers are implemented as RAZ.
- All RNI and RND nodes control three interface ports. The other variants that control one or two ports are not supported.
- QoS is not supported and all related registers are RAZ/WI.
- Error injection and error generation are not supported. All error registers are RAZ/WI.
- Power/Clock/Interrupt signals are not supported.
- RN SAM and HN-F SAM memory partition size of 128MB is not supported.
- HNT nodes are not supported.
- Atomic operations to noncachable/device regions are not supported.
- External RNSAM is not supported, except based on node type. The model populates the relevant bit in the child pointer register.
- By default, HNF hashing uses the address[12:MAX] instead of the actual address[6:MAX], due to the DMI mechanism in the model. Enable the parameter `enable_snf_hashing` to make hashing logic use the actual address[6:MAX].
- 0KB SLC data RAM and tag RAM are not supported.
- There are no constraints on mesh sizing when XID_WIDTH and YID_WIDTH = 4.
- The CML feature is supported at beta quality.

About the debug_force_snoop parameter

The CMN interconnect normally starts up with snooping disabled.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

The parameters `rnf_sci_enable` and `rni_sci_enable` determine which connections are managed by the System Coherency Interface. Connections that are managed by the System Coherency Interface look after themselves for entering and leaving the coherency domain and `debug_force_snoop` has no effect on those ports.

However, for connections that are not managed by the System Coherency Interface, `debug_force_snoop` enables the system upstream of a port to be snooped if the upstream is not in reset and if `acchannelen_rnf` allows it.

Therefore, software does not need to know how to turn snoops in the interconnect on and off. The reset state of the upstream rnf ports is determined by the `rnf_upstream_reset_state[]` signals and must be connected for any port that could go into reset and you want managed by `debug_force_snoop`.

If the connection is managed by SCI or can never go into reset then you need not connect this and the interconnect assumes that the upstream system is not in reset and will always enable the snoops if `ACCHANNELENSx` allows it.

If you fail to connect `rnf_upstream_reset_state[]` correctly, then the upstream system might receive snoop messages while in reset and it will complain that it received a snoop request while it was in reset.

If software overrides the enables then the next change to `rnf_upstream_reset_state[]` or the interconnect reset will overwrite the software values.

Using this operation along with software controlling the enables could confuse the software. This option does not remove the responsibility of the upstream system to clean any shared dirty data from its caches before going into reset.

Note

This parameter is for debug purposes only. Do not use it instead of correctly configuring CMN.

Table 3-334 Ports

Name	Protocol	Type	Description
`event_clusters`	*Signal* on page 2-85	Peer	CPU event communication signal from the clusters.
`event_downstream_link_signal[6]`	*Signal* on page 2-85	Master	CPU event communication signal to the CMLHub.
`event_upstream_link_signal[6]`	*Signal* on page 2-85	Slave	-
`pvbus_m_cml`	`PVBus`	Master	CML downstream ports
`pvbus_m_cml_cfg`	`PVBus`	Master	CML downstream hub configuration port
`pvbus_m_hni[8]`	`PVBus`	Master	HNI downstream ports.
`pvbus_m_snf[16]`	`PVBus`	Master	SNF downstream port.
`pvbus_s_cml`	`PVBus`	Slave	CML upstream ports
`pvbus_s_rnf[64]`	`PVBus`	Slave	RNF upstream ports.
`pvbus_s_rni[96]`	`PVBus`	Slave	RNI upstream ports.
`reset_in`	*Signal* on page 2-85	Slave	Reset signal.
`rnf_sci_s[64]`	`SystemCoherencyInterface`	Slave	System Coherency port to move the RNF master in and out of the coherency domain.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-334 Ports (continued)

Name	Protocol	Type	Description
rnf_upstream_reset_in[64]	*Signal* on page 2-85	Slave	Used by the interconnect to determine the reset state of the RNF master to check an aspect of reset sequencing. If debug_force_snoop is used, this port must be connected.
rni_sci_s[96]	SystemCoherencyInterface	Slave	System Coherency port to move the RNI master in and out of the coherency domain.
rni_upstream_reset_in[96]	*Signal* on page 2-85	Slave	Used by the interconnect to determine the reset state of the RNI master to check an aspect of reset sequencing. If debug_force_snoop is used, this port must be connected.

Table 3-335 Parameters for CMN600Cache

Name	Type	Default value	Description
cmn600_cache.cache_state_modelled	bool	0x1	Model the cache state.
cmn600_cache.force_on_from_start	bool	0x0	The CMN600 will normally start up with snooping disabled, however, using this then we allow the model to start up as enabled without having to program it. This is only setup at simulation reset and not at signal reset. If the upstreams can ever be held in reset then you *must* connect the reset_state_of_upstream_port so that it knows when to disable snoops to the upstream systems. Otherwise, the upstream system will receive snoop messages whilst in reset and it will complain that it 'received a snoop request whilst it was in reset'.
cmn600_cache.number_of_phantom_entries	int	0x20	Number of phantom entries in the cache. Phantom entries are used by certain cache operations to hold temporary data. Usually this should be left at the default value which is safe for all systems containing up to 32 masters.
cmn600_cache.reentrancy_support	string	"env"	Must be one of: on/off/cacheglobal/env: 'on': hazard checking per cache line (normal mode), 'off': no hazard checking (use only for single master systems), 'cacheglobal': hazard checking globally for cache (not per cache line, testing feature, provokes more hazards than necessary), 'env' (or empty string): take value from FM_REENTRANCY_SUPPORT env var, if this is not set use 'on', default is 'env'

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-336 Parameters for CMN600

Name	Type	Default value	Description
acchannelen_rnf	string	"0"	For each rnf port, indicates if the port is populated with a snoop responding device or not The input value is a string, for example '0xffff' or 'ffff'
acchannelen_rni	string	"0"	For each rni port, indicates if the port is populated with a dvm responding device or not. The input value is a string, for example '0xffff' or 'ffff'
cache_state_modelled	bool	0x0	Model the cache state.
debug_force_snoop	bool	0x0	The CMN600 interconnect will normally start with snooping disabled. The parameter rnf_sci_enable and rni_sci_enable determines which connections are managed by the System Coherency Interface. For connections that are managed by the System Coherency Interface, then they will look after themselves for entering and leaving the coherency domain and this parameter has no effect for those ports. However, for connections that are _not_ managed by the System Coherency Interface, this parameter enables the upstream system of a port to be snooped if the upstream is not in reset and if acchannelen_rnf allows it. NOTE: This parameter is for debug purpose only Do not use this parameter instead of correctly configuring CMN. Refer CMN600 documentation page for more info about this parameter in fast models reference manual
default_hni_port	int	0xffff ffff	HNI port number that is the default boot target.
enable_logger	bool	0x0	Enable PVBusLoggers for the downstream ports in the CMN model.
enable_rnsam_to_hnf_wider_hash	bool	0x0	Enable support of wider hash for the RNSAM to HNF communication. If this variable enabled, then bits[47:6] from PA used in hashing function. By default it is [47:12].
force_rnsam_internal	bool	0x1	Force all RNSAMs to be internal independently of the mesh topology.
hnf_cache_size	string	"128KB"	HNF Node Cache size in KB/MB e.g. 128KB, 2MB
hni_index_to_nodeid	string	""	A string parameter to define relation between hni port num and hni nodeid, for example 0=0x14,1=0x48
mesh_config_file	string	""	Name of a file containing mesh placement of cmn600 components
periphbase	int	0x2c00 0000	Value for PERIPHBASE. Bits [25:0] are treated 0
print_cmn_ccix_config	bool	0x0	Print information about the CCIX configuration.
print_cmn_config	bool	0x0	Print the mesh topology and children pointers acquired from the YML file.
revision	string	"r1p1"	Component revision. Currently r1p1 and r3p0 are supported.
rnd_index_to_nodeid	string	""	A string parameter to define relation between rnd port num and rnd nodeid, for example 0=0x14,1=0x48. Each rnd node covers 3 ports, so port num is multiplied by 3 to cover 3 consecutive ports. So, the ports assigned will be 0,1,2=0x14 and 3,4,5=0x48

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-336 Parameters for CMN600 (continued)

Name	Type	Default value	Description
rnf_index_to_nodeid	string	""	A string parameter to define relation between rnf port num and rnf nodeid, for example 0=0x14,1=0x48
rnf_sci_enable	string	"0x0"	For each rnf port, indicates if the port is managed by System Coherency Interface for coherency domain entry/exit 1 = Managed by SCI, 0 = Managed by Software. The input value is a string, for example '0xffff' or 'ffff'
rni_index_to_nodeid	string	""	A string parameter to define relation between rni port num and rni nodeid, for example 0=0x14,1=0x48. Each rni node covers 3 ports, so port num is multiplied by 3 to cover 3 consecutive ports. So, the ports assigned will be 0,1,2=0x14 and 3,4,5=0x48
rni_sci_enable	string	"0x0"	For each rni port, indicates if the port is managed by System Coherency Interface for coherency domain entry/exit 1 = Managed by SCI, 0 = Managed by Software. The input value is a string, for example '0xffff' or 'ffff'
sbsx_index_to_nodeid	string	""	A string parameter to define relation between sbsx port num and sbsx nodeid, for example 0=0x14,1=0x48
skip_cmn_config_check	bool	0x0	Skip any topology configuration checks. The maximum number of devices per type not verified.
snf_index_to_nodeid	string	""	A string parameter to define relation between snf port num and snf nodeid, for example 0=0x14,1=0x48

3.9.12 CMN600CMLHub

CMN600 CML Interconnect Hub Fast Model. This model is written in C++ and models version r1p1 of the RTL.

CMN600CMLHub contains the following CADI targets:

- CMN600

CMN600CMLHub contains the following MTI components:

- *4.57 CMN600CML - trace* on page 4-2428
- *4.102 PVBusExclusiveMonitor - trace* on page 4-2752
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.108 PVCache - trace* on page 4-2761

Table 3-337 Ports

Name	Protocol	Type	Description
event_downstream_link_signal[4]	*Signal* on page 2-85	Slave	CPU event communication signal.
event_upstream_link_signal[4]	*Signal* on page 2-85	Master	-
pvbus_m[4]	PVBus	Master	Downstream CCIX port.
pvbus_s[4]	PVBus	Slave	Upstream CCIX ports.
pvbus_s_cfg[4]	PVBus	Slave	Upstream config ports.
reset_signal	*Signal* on page 2-85	Slave	Reset signal.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-338 Parameters for CMN600

Name	Type	Default value	Description
acchannelen_rnf	string	"0"	For each rnf port, indicates if the port is populated with a snoop responding device or not The input value is a string, for example '0xffff' or 'ffff'
acchannelen_rni	string	"0"	For each rni port, indicates if the port is populated with a dvm responding device or not. The input value is a string, for example '0xffff' or 'ffff'
cache_state_modelled	bool	0x0	Model the cache state.
debug_force_snoop	bool	0x0	The CMN600 interconnect will normally start with snooping disabled. The parameter rnf_sci_enable and rni_sci_enable determines which connections are managed by the System Coherency Interface. For connections that are managed by the System Coherency Interface, then they will look after themselves for entering and leaving the coherency domain and this parameter has no effect for those ports. However, for connections that are _not_ managed by the System Coherency Interface, this parameter enables the upstream system of a port to be snooped if the upstream is not in reset and if acchannelen_rnf allows it. NOTE: This parameter is for debug purpose only Do not use this parameter instead of correctly configuring CMN. Refer CMN600 documentation page for more info about this parameter in fast models reference manual
default_hni_port	int	0xffff ffff	HNI port number that is the default boot target.
enable_logger	bool	0x0	Enable PVBusLoggers for the downstream ports in the CMN model.
enable_rnsam_to_hnf_wider_hash	bool	0x0	Enable support of wider hash for the RNSAM to HNF communication. If this variable enabled, then bits[47:6] from PA used in hashing function. By default it is [47:12].
force_rnsam_internal	bool	0x1	Force all RNSAMs to be internal independently of the mesh topology.
hnf_cache_size	string	"128KB"	HNF Node Cache size in KB/MB e.g. 128KB, 2MB
hni_index_to_nodeid	string	""	A string parameter to define relation between hni port num and hni nodeid, for example 0=0x14,1=0x48
mesh_config_file	string	""	Name of a file containing mesh placement of cmn600 components
periphbase	int	0x2c00 0000	Value for PERIPHBASE. Bits [25:0] are treated 0
print_cmn_ccix_config	bool	0x0	Print information about the CCIX configuration.
print_cmn_config	bool	0x0	Print the mesh topology and children pointers acquired from the YML file.
revision	string	"r1p1"	Component revision. Currently r1p1 and r3p0 are supported.
rnd_index_to_nodeid	string	""	A string parameter to define relation between rnd port num and rnd nodeid, for example 0=0x14,1=0x48. Each rnd node covers 3 ports, so port num is multiplied by 3 to cover 3 consecutive ports. So, the ports assigned will be 0,1,2=0x14 and 3,4,5=0x48

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-338 Parameters for CMN600 (continued)

Name	Type	Default value	Description
rnf_index_to_nodeid	string	""	A string parameter to define relation between rnf port num and rnf nodeid, for example 0=0x14,1=0x48
rnf_sci_enable	string	"0x0"	For each rnf port, indicates if the port is managed by System Coherency Interface for coherency domain entry/exit 1 = Managed by SCI, 0 = Managed by Software. The input value is a string, for example '0xffff' or 'ffff'
rni_index_to_nodeid	string	""	A string parameter to define relation between rni port num and rni nodeid, for example 0=0x14,1=0x48. Each rni node covers 3 ports, so port num is multiplied by 3 to cover 3 consecutive ports. So, the ports assigned will be 0,1,2=0x14 and 3,4,5=0x48
rni_sci_enable	string	"0x0"	For each rni port, indicates if the port is managed by System Coherency Interface for coherency domain entry/exit 1 = Managed by SCI, 0 = Managed by Software. The input value is a string, for example '0xffff' or 'ffff'
sbsx_index_to_nodeid	string	""	A string parameter to define relation between sbsx port num and sbsx nodeid, for example 0=0x14,1=0x48
skip_cmn_config_check	bool	0x0	Skip any topology configuration checks. The maximum number of devices per type not verified.
snf_index_to_nodeid	string	""	A string parameter to define relation between snf port num and snf nodeid, for example 0=0x14,1=0x48

3.9.13 CMN650

CMN650 Interconnect Fast Model. This model is written in C++ and models version r1p1 of the RTL.

CMN650 contains the following CADI targets:

- CMN650

CMN650 contains the following MTI components:

- *4.58 CMN600Cache - trace* on page 4-2429
- *4.59 CMN650 - trace* on page 4-2434
- *4.102 PVBusExclusiveMonitor - trace* on page 4-2752
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760

About CMN650

The CMN650 model supports the following:

- rnf, rni/rnd, hni, and snf/sbsx interface ports. The mapping between the port number and `NodeId` is based on the `NodeId` index. For example, RNF2 controls `pvbus_s_rnf[2]`. Its index is specified in the `node_info` register as `logical_id`. Similar behavior can be expected for HNI.

  If both RND and RNI nodes are present, then all starting rni ports are mapped to RND nodes and then the RNI nodes. For example, for CMN650 with two RND nodes, one RNI node, and given each RNI/RND node controls three interface ports, `pvbus_s_rni[0-2]` maps to RND0, `pvbus_s_rni[3-5]` maps to RND1 and `pvbus_s_rni[6-8]` maps to RNI0.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Similarly, SNF and SBSX nodes are mapped to `pvbus_m_snf[]` ports where starting ports are mapped to SNF and then SBSX nodes.

- The parameter `mesh_config_file` is used to define the mesh placement of the CHI nodes in the following ways:
  - This is recommended. Set `mesh_config_file` to the name of the yml configuration file emitted by Socrates.
  - This is deprecated. Set `mesh_config_file` to a JSON file, with a `.json` file extension. The JSON file has the following format:

```
{
    "mesh_row_size":4,
    "mesh_column_size":3,
    "mesh_placement" : {
    "0.0" : "RND:HND",  "1.0" : "RND:RNF",  "2.0" : "RND:RNF",
    "0.1" : "HNF:HNF",  "1.1" : "HNF:HNF",  "2.1" : "HNF:HNF",
    "0.2" : "HNF:HNF",  "1.2" : "HNF:HNF",  "2.2" : "HNF:HNF",
    "0.3" : "RND:HNF",  "1.3" : "HNF:HNF",  "2.3" : "HNF:SNF"
    }
}
```

    In the `mesh_placement` tag:
    - The value to the left of a colon (`:`) indicates the position of the XP. For example, `"2.3"` means the XP is positioned at coordinates X=2 and Y=3.
    - The value to the right of a colon (`:`) indicates the node protocols for port 0 and port 1. For example, `"RND:HND"` means RND is attached to port 0 and HND is attached to port 1.
- The discovery feature to determine the system address of CHI nodes.
- Hashed and non-hashed memory regions, and the `sys_cache_group` for hashed memory regions.
- DVM filtering using VMF registers.
- Entry/exit of RNs from snoop and DVM domains using both `SYSCOREQ`/`SYSCOACK` signals and through software using MXP `syscoreq` registers.
- Reading the content of the cache through HNF `l3_cache_access*` registers, if `cache_state_modelled` is true.
- The HN-F SAM feature to allocate a range of addresses to a specific SNF.
- HN-F cache way lock and OCM.
- The CAL (Core Aggregation Layer) feature.
- The CML (Coherent Multichip Link) feature.
- Use the `print_cmn_ccix_config` parameter to print to `stdout` details of the mesh configuration and memory map for the model.

Model limitations

- PMU counters are not supported. Counter registers are implemented as RAZ.
- All RNI and RND nodes control three interface ports. The other variants that control one or two ports are not supported.
- QoS is not supported and all related registers are RAZ/WI.
- Error injection and error generation are not supported. All error registers are RAZ/WI.
- Power, clock, and interrupt signals are not supported.
- RN SAM and HN-F SAM memory partition size of 128MB is not supported.
- HNT nodes are not supported.
- External RNSAM is not supported, except based on node type. The model populates the relevant bit in the child pointer register.
- By default, HNF hashing uses the address[12:MAX] instead of the actual address[6:MAX], due to the DMI mechanism in the model. Enable the parameter `enable_snf_hashing` to make hashing logic use the actual address[6:MAX].
- 0KB SLC data RAM and tag RAM are not supported.
- HNF-SNF hashing is not supported. HNSAM SNF hashing is a single memory range mapped to multiple SNFs. Example HNFSAM_Mem_Map:

```
0x0000001000 - 0x0000002000 SNF0 SNF1
```

  In this case, the interconnect hardware should hash this range between two SNFs, similarly to HNF hashing. In this release, the model does not support this SNF hashing.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SNF mapping in HNFSAM is supported. Example HNFSAM_Mem_Map:

```
0x0000001000 - 0x0000002000 SNF0
0x0000002000 - 0x0000003000 SNF1
```

About the debug_force_snoop parameter

The CMN interconnect normally starts up with snooping disabled.

The parameters `rnf_sci_enable` and `rni_sci_enable` determine which connections are managed by the System Coherency Interface. Connections that are managed by the System Coherency Interface look after themselves for entering and leaving the coherency domain and `debug_force_snoop` has no effect on those ports.

However, for connections that are not managed by the System Coherency Interface, `debug_force_snoop` enables the system upstream of a port to be snooped if the upstream is not in reset and if `acchannelen_rnf` allows it.

Therefore, software does not need to know how to turn snoops in the interconnect on and off. The reset state of the upstream rnf ports is determined by the `rnf_upstream_reset_state[]` signals and must be connected for any port that could go into reset and you want managed by `debug_force_snoop`.

If the connection is managed by SCI or can never go into reset then you need not connect this and the interconnect assumes that the upstream system is not in reset and will always enable the snoops if `ACCHANNELENSx` allows it.

If you fail to connect `rnf_upstream_reset_state[]` correctly, then the upstream system might receive snoop messages while in reset and it will complain that it received a snoop request while it was in reset.

If software overrides the enables then the next change to `rnf_upstream_reset_state[]` or the interconnect reset will overwrite the software values.

Using this operation along with software controlling the enables could confuse the software. This option does not remove the responsibility of the upstream system to clean any shared dirty data from its caches before going into reset.

Note

This parameter is for debug purposes only. Do not use it instead of correctly configuring CMN.

Table 3-339 Ports

Name	Protocol	Type	Description
`event_clusters`	*Signal* on page 2-85	Peer	CPU event communication signal from the clusters.
`event_downstream_link_signal[10]`	*Signal* on page 2-85	Master	CPU event communication signal to the CMLHub.
`event_upstream_link_signal[10]`	*Signal* on page 2-85	Slave	-
`pvbus_m_cml`	`PVBus`	Master	CML downstream ports
`pvbus_m_cml_cfg`	`PVBus`	Master	CML downstream hub configuration port
`pvbus_m_hni[16]`	`PVBus`	Master	HNI downstream ports.
`pvbus_m_snf[40]`	`PVBus`	Master	SNF downstream port.
`pvbus_s_apb`	`PVBus`	Slave	APB interface port.
`pvbus_s_cml`	`PVBus`	Slave	CML upstream ports
`pvbus_s_rnf[64]`	`PVBus`	Slave	RNF upstream ports.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-339 Ports (continued)

Name	Protocol	Type	Description
pvbus_s_rni[96]	PVBus	Slave	RNI upstream ports.
reset_in	*Signal* on page 2-85	Slave	Reset signal.
rnf_sci_s[64]	SystemCoherencyInterface	Slave	System Coherency port to move the RNF master in and out of the coherency domain.
rnf_upstream_reset_in[64]	*Signal* on page 2-85	Slave	Used by the interconnect to determine the reset state of the RNF master to check an aspect of reset sequencing. If debug_force_snoop is used, this port must be connected.
rni_sci_s[96]	SystemCoherencyInterface	Slave	System Coherency port to move the RNI master in and out of the coherency domain.
rni_upstream_reset_in[96]	*Signal* on page 2-85	Slave	Used by the interconnect to determine the reset state of the RNI master to check an aspect of reset sequencing. If debug_force_snoop is used, this port must be connected.

Table 3-340 Parameters for CMN650

Name	Type	Default value	Description
acchannelen_rnf	string	"0"	For each rnf port, indicates if the port is populated with a snoop responding device or not The input value is a string, for example '0xffff' or 'ffff'
acchannelen_rni	string	"0"	For each rni port, indicates if the port is populated with a dvm responding device or not. The input value is a string, for example '0xffff' or 'ffff'
cache_state_modelled	bool	0x0	Model the cache state.
debug_force_snoop	bool	0x0	The CMN650 interconnect will normally start with snooping disabled. The parameter rnf_sci_enable and rni_sci_enable determines which connections are managed by the System Coherency Interface. For connections that are managed by the System Coherency Interface, then they will look after themselves for entering and leaving the coherency domain and this parameter has no effect for those ports. However, for connections that are _not_ managed by the System Coherency Interface, this parameter enables the upstream system of a port to be snooped if the upstream is not in reset and if acchannelen_rnf allows it. NOTE: This parameter is for debug purpose only Do not use this parameter instead of correctly configuring CMN. Refer CMN650 documentation page for more info about this parameter in fast models reference manual
default_hni_port	int	0xffff ffff	HNI port number that is the default boot target.
enable_logger	bool	0x0	Enable PVBusLoggers for the downstream ports in the CMN model.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-340 Parameters for CMN650 (continued)

Name	Type	Default value	Description
enable_rnsam_to_hnf_wider_hash	bool	0x0	Enable support of wider hash for the RNSAM to HNF communication. If this variable enabled, then bits[47:6] from PA used in hashing function. By default it is [47:12].
force_rnsam_internal	bool	0x1	Force all RNSAMs to be internal independently of the mesh topology.
hnf_cache_size	string	"128KB"	HNF Node Cache size in KB/MB following HNF_unit_info register, accepted values: 0, 128KB, 256KB, 512KB, 1MB, 2MB, 3MB, and 4 MB
hni_index_to_nodeid	string	""	A string parameter to define relation between hni port num and hni nodeid, for example 0=0x14,1=0x48
mesh_config_file	string	""	Name of a file containing mesh placement of cmn650 components
periphbase	int	0x2000 0000	Value for PERIPHBASE. Bits [27:0] are treated 0
print_cmn_ccix_config	bool	0x0	Print information about the CCIX configuration.
print_cmn_config	bool	0x0	Print the mesh topology and children pointers acquired from the YML file.
revision	string	"r1p1"	Component revision. Currently r1p1 is supported.
rnd_index_to_nodeid	string	""	A string parameter to define relation between rnd port num and rnd nodeid, for example 0=0x14,1=0x48. Each rnd node covers 3 ports, so port num is multiplied by 3 to cover 3 consecutive ports. So, the ports assigned will be 0,1,2=0x14 and 3,4,5=0x48
rnf_index_to_nodeid	string	""	A string parameter to define relation between rnf port num and rnf nodeid, for example 0=0x14,1=0x48
rnf_sci_enable	string	"0x0"	For each rnf port, indicates if the port is managed by System Coherency Interface for coherency domain entry/exit 1 = Managed by SCI, 0 = Managed by Software. The input value is a string, for example '0xffff' or 'ffff'
rni_index_to_nodeid	string	""	A string parameter to define relation between rni port num and rni nodeid, for example 0=0x14,1=0x48. Each rni node covers 3 ports, so port num is multiplied by 3 to cover 3 consecutive ports. So, the ports assigned will be 0,1,2=0x14 and 3,4,5=0x48
rni_sci_enable	string	"0x0"	For each rni port, indicates if the port is managed by System Coherency Interface for coherency domain entry/exit 1 = Managed by SCI, 0 = Managed by Software. The input value is a string, for example '0xffff' or 'ffff'
sbsx_index_to_nodeid	string	""	A string parameter to define relation between sbsx port num and sbsx nodeid, for example 0=0x14,1=0x48
skip_cmn_config_check	bool	0x0	Skip any topology configuration checks. The maximum number of devices per type not verified.
snf_index_to_nodeid	string	""	A string parameter to define relation between snf port num and snf nodeid, for example 0=0x14,1=0x48
yml_has_node_addresses	bool	0x0	Does the top-level YML file describe node-addresses ?

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

3.9.14 DMC500

ARM Dynamic Memory Controller(DMC500). This model is written in C++.

DMC500 contains the following CADI targets:

- DMC500

DMC500 contains the following MTI components:

- *4.62 DMC-500 - trace* on page 4-2448
- *4.107 PVBusSlave - trace* on page 4-2760

About DMC500

A platform can have multiple instances of this component. For example:

```
//LISA instantiation
composition
{
    // Memory controllers
    dmc0    : DMC500("default_region_attributes"=dmc_default_region_attributes,
                     "default_region_id_access"=dmc_default_region_id_access,
                     "passthrough_debug_access"=true);
    dmc1    : DMC500("default_region_attributes"=dmc_default_region_attributes,
                     "default_region_id_access"=dmc_default_region_id_access,
                     "passthrough_debug_access"=true);
}
```

Differences between the model and the RTL

The model has the following limitations:

- It does not support address striping.
- It works with linear addresses and not in rank,bank,row,column form.
- It does not include any mechanism for error injection or detection.
- Scrubbing functionality is only provided from the interface point of view.
- It does not implement direct read or write commands.
- It does not implement any performance counters.
- All OR'd interrupt signals are missing from this release of the model. Users can connect the failed access interrupt as a substitute.
- The model combines separate failed access interrupts for system interfaces 1 and 2 into a single failed access interrupt.
- DMC-500 has three separate reset signals whereas this model has a single reset signal which supports the combined assertion of three resets. This model does not support separate reset signals.

Table 3-341 Ports

Name	Protocol	Type	Description
apb_pvbus_s	PVBus	Slave	Programmers interface to program and control the DMC-500.
failed_access_interrupt_signal	*Signal* on page 2-85	Master	The DMC has detected a system request that has failed a permissions check and a previously detected assertion was not cleared.
filter_pvbus_m	PVBus	Master	DMC master port from System Interface 0 to memory.
filter_pvbus_s	PVBus	Slave	System interface 0. Generally, Non-coherent Interface.
reset_signal	*Signal* on page 2-85	Slave	DMC reset.
si1_filter_pvbus_m	PVBus	Master	DMC master port from System Interface 1 to memory.
si1_filter_pvbus_s	PVBus	Slave	System interface 1. Generally, Coherent Interface.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

3.9.15 DMC520

ARM Dynamic Memory Controller(DMC520). This model is written in C++.

DMC520 contains the following CADI targets:

- DMC520

DMC520 contains the following MTI components:

- *4.63 DMC-520 - trace* on page 4-2450
- *4.107 PVBusSlave - trace* on page 4-2760

About DMC520

A platform can have multiple instances of this component. For example:

```
//LISA instantiation
composition
{
    // Memory controllers
    dmc520_0            : DMC520("passthrough_debug_access"=true);
    dmc520_1            : DMC520("passthrough_debug_access"=true);
}
```

Limitations:

- The model does not support address striping.
- It works with linear addresses and not in rank,bank,row,column form.
- It does not include any mechanism for error injection or detection.
- Scrubbing functionality is only provided from the interface point of view.
- It does not implement direct read or write commands.
- It does not implement any performance counters.

Differences between the model and the RTL

The DMC520 and DMC620 models have different interfaces to those in the hardware due to the level of abstraction of memory in Fast Models. These are the differences:

- Like the hardware, the model has a slave port for configuring register accesses, `apb_pvbus_s`, and an AXI interface for incoming memory transactions that are attempting to access memory that is managed by the DMC.
- The hardware component translates incoming transactions on the AXI interface to a format that is conducive to accessing DRAM chips. The model performs TrustZone access control and models the DMC readiness state, but does not translate the transactions. If allowed, the model forwards incoming transactions to be handled by a memory storage handling component that works at the transaction level.

Table 3-342 Ports

Name	Protocol	Type	Description
`all_or_interrupt_signal`	*Signal* on page 2-85	Master	A combined interrupt that is the logical OR of the other interrupts.
`apb_pvbus_s`	`PVBus`	Slave	Programmers interface to program and control the DMC-520.
`arch_fsm_interrupt_signal`	*Signal* on page 2-85	Master	The DMC has detected a change in the architectural state.
`failed_access_interrupt_signal`	*Signal* on page 2-85	Master	The DMC has detected a system request that has failed a permissions check and a previously detected assertion was not cleared.
`filter_pvbus_m`	`PVBus`	Master	DMC master port to memory.
`filter_pvbus_s`	`PVBus`	Slave	System interface.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-342 Ports (continued)

Name	Protocol	Type	Description
reset_signal	*Signal* on page 2-85	Slave	DMC reset.
scrub_event_in[8]	*Signal* on page 2-85	Slave	Scrub event n trigger.
scrub_event_out[8]	*Signal* on page 2-85	Master	Scrub event n triggered.

3.9.16 DMC620

ARM Dynamic Memory Controller(DMC620). This model is written in C++.

DMC620 contains the following CADI targets:

- DMC620

DMC620 contains the following MTI components:

- *4.64 DMC-620 - trace* on page 4-2452
- *4.107 PVBusSlave - trace* on page 4-2760

About DMC620

The model has the following limitations:

- It does not support address striping.
- It works with linear addresses and not in rank,bank,row,column form.
- It includes error injection and detection mechanisms and syndrome registers support only for RAS error types 4 (ECC single-bit SRAM error) and 5 (ECC double-bit SRAM error).
- Scrubbing functionality is not provided.
- It does not implement direct read or write commands.
- It does not implement any performance counters.

Differences between the model and the RTL

The DMC520 and DMC620 models have different interfaces to those in the hardware due to the level of abstraction of memory in Fast Models. These are the differences:

- Like the hardware, the model has a slave port for configuring register accesses, apb_pvbus_s, and an AXI interface for incoming memory transactions that are attempting to access memory that is managed by the DMC.
- The hardware component translates incoming transactions on the AXI interface to a format that is conducive to accessing DRAM chips. The model performs TrustZone access control and models the DMC readiness state, but does not translate the transactions. If allowed, the model forwards incoming transactions to be handled by a memory storage handling component that works at the transaction level.

Table 3-343 Ports

Name	Protocol	Type	Description
all_or_interrupt_signal	*Signal* on page 2-85	Master	A combined interrupt that is the logical OR of the other interrupts.
apb_pvbus_s	PVBus	Slave	Programmers interface to program and control the DMC-620.
arch_fsm_interrupt_signal	*Signal* on page 2-85	Master	The DMC has detected a change in the architectural state.
failed_access_interrupt_signal	*Signal* on page 2-85	Master	The DMC has detected a system request that has failed a permissions check and a previously detected assertion was not cleared.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-343 Ports (continued)

Name	Protocol	Type	Description
filter_pvbus_m	PVBus	Master	DMC master port to memory.
filter_pvbus_s	PVBus	Slave	System interface.
interrupt_cfh_master	*Signal* on page 2-85	Master	The DMC has detected and corrected a single bit error on the RAM access.
interrupt_combined_oflow_master	*Signal* on page 2-85	Master	The DMC has detected a counter overflow.
interrupt_fh_master	*Signal* on page 2-85	Master	The DMC has detected a double bit error on the RAM access.
reset_signal	*Signal* on page 2-85	Slave	DMC reset.

3.9.17 DMC_400

ARM PrimeCell Dynamic Memory Controller(DMC400). This model is written in LISA+.

DMC_400 contains the following CADI targets:

- DMC_400

DMC_400 contains the following MTI components:

- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760

About DMC_400

The configuration of this model by setting the registers does not generally affect accesses to main memory.

This component has no timing information, so changing the values of the timing registers has no effect on behavior. The memory models do not attach to the component, and error checking does not update registers because the model does not include the possibility of errors.

Table 3-344 Ports

Name	Protocol	Type	Description
apb_interface	PVBus	Slave	Slave bus interface for register access.
axi_if_in[4]	PVBus	Slave	Slave bus for connecting to bus decoder.
axi_if_out[4]	PVBus	Master	Master to connect to DRAM.
clr_ex_mon	*Signal* on page 2-85	Master	Indicates when global monitors state is cleared.
user_status_ext	*Value* on page 2-85	Slave	Allow user status to be set from outside.

Table 3-345 Parameters for DMC_400

Name	Type	Default value	Description
ECC_SUPPORT	bool	0x1	Does the controller support ECC?
IF_CHIP0	int	-0x1	Set this parameter to 0 if memory is connected
IF_CHIP1	int	-0x1	Set this parameter to 0 if memory is connected
IF_CHIP2	int	-0x1	Set this parameter to 0 if memory is connected
IF_CHIP3	int	-0x1	Set this parameter to 0 if memory is connected

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-345 Parameters for DMC_400 (continued)

Name	Type	Default value	Description
MEMORY_WIDTH	int	0x20	Valid widths are 16, 32 or 64 bits
diagnostics	int	0x0	Diagnostics
revision_string	string	"r0p1"	Revision

3.9.18 EthosU55

Arm Ethos-U55 microNPU. This model is written in C++.

About EthosU55

To use this component, include the following line in your sgrepo file:

```
path = "$(PVLIB_HOME)/LISA/ethosu.sgrepo";
```

The num_macs parameter controls the number of 8x8 MACs performed per cycle. The valid values are 32, 64, 128, or 256. The default value is 128.

Limitations

This model does not support Windows hosts or GCC 4.9. It only supports Linux hosts and GCC 6.4 or GCC 7.3.

Table 3-346 Ports

Name	Protocol	Type	Description
clk_in	ClockSignal	Slave	NPU clock signal.
irq_out	*Signal* on page 2-85	Master	Sends interrupt requests to the external host application processor.
pvbus_m0	PVBus	Master	Master port 0 to access external memory.
pvbus_m1	PVBus	Master	Master port 1 to access external memory.
pvbus_s	PVBus	Slave	Slave port to access the control registers of the NPU.
resetn_in	*Signal* on page 2-85	Slave	NPU reset signal.

3.9.19 GIC500

GIC500 Component for distribution of interrupts. This model is written in C++.

GIC500 contains the following CADI targets:

- GIC500

GIC500 contains the following MTI components:

- *4.76 GICv3Distributor - trace* on page 4-2496
- *4.77 GICv3IRI - trace* on page 4-2504
- *4.78 GICv3InterruptTranslationService - trace* on page 4-2505
- *4.80 GICv3Redistributor - trace* on page 4-2527
- *4.81 GICv3RedistributorInternal - trace* on page 4-2541

About GIC500

This is a single-component implementation of the GICv3 architecture with support for 256 cores. You can configure the model to support a maximum of 32 clusters with 8 cores per cluster. Use it with an Armv8-A core to deliver interrupts. It supports a single Interrupt Translation Service for message-based interrupts. It supports the architectural features, but does not support the IMPLEMENTATION DEFINED features.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

You must configure some parameters in order to use the GIC500 component. For example:

```
gic500: GIC500(
    "num_clusters" = 2,
    "cpus_per_cluster_0" = 4,
    "cpus_per_cluster_1" = 4,
    "reg-base" = 0x2c200000,
    "SPI-count" = 256
);
```

Note

- Set the `FASTSIM_GIC_MEMORY_MAP` environment variable to 1 to print to `stderr` the memory map of any GICv3 or later models that are included in the platform being run.
- For writes to `GITS_TRANSLATE64R`, the *MSIRewriter* on page 3-104 component can be used to create the correct data transactions based on the device ID.

MasterID, ExtendedID, and UserFlags

This table shows how this model encodes information in the `MasterID`, `ExtendedID`, and `UserFlags` bus attributes:

Table 3-347 pvbus_s port

PVBus attribute	Bits used	Property encoded	Notes
`MasterID`	Bits[31:0]	Master ID that invoked the register access	-
`ExtendedID`	-	-	Not used.
`UserFlags`	-	-	Not used.

Note

The `pvbus_m` port does not use `MasterID`, `ExtendedID`, or `UserFlags`.

Table 3-348 Ports

Name	Protocol	Type	Description
`cfgsdisable`	*Signal* on page 2-85	Slave	Disable some SPIs signal.
`cpu_active_0[8]`	*Signal* on page 2-85	Slave	cpu_active pins of cluster 0.
`cpu_active_1[8]`	*Signal* on page 2-85	Slave	cpu_active pins of cluster 1.
`cpu_active_10[8]`	*Signal* on page 2-85	Slave	cpu_active pins of cluster 10.
`cpu_active_11[8]`	*Signal* on page 2-85	Slave	cpu_active pins of cluster 11.
`cpu_active_12[8]`	*Signal* on page 2-85	Slave	cpu_active pins of cluster 12.
`cpu_active_13[8]`	*Signal* on page 2-85	Slave	cpu_active pins of cluster 13.
`cpu_active_14[8]`	*Signal* on page 2-85	Slave	cpu_active pins of cluster 14.
`cpu_active_15[8]`	*Signal* on page 2-85	Slave	cpu_active pins of cluster 15.
`cpu_active_16[8]`	*Signal* on page 2-85	Slave	cpu_active pins of cluster 16.
`cpu_active_17[8]`	*Signal* on page 2-85	Slave	cpu_active pins of cluster 17.
`cpu_active_18[8]`	*Signal* on page 2-85	Slave	cpu_active pins of cluster 18.
`cpu_active_19[8]`	*Signal* on page 2-85	Slave	cpu_active pins of cluster 19.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-348 Ports (continued)

Name	Protocol	Type	Description
cpu_active_2[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 2.
cpu_active_20[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 20.
cpu_active_21[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 21.
cpu_active_22[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 22.
cpu_active_23[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 23.
cpu_active_24[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 24.
cpu_active_25[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 25.
cpu_active_26[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 26.
cpu_active_27[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 27.
cpu_active_28[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 28.
cpu_active_29[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 29.
cpu_active_3[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 3.
cpu_active_30[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 30.
cpu_active_31[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 31.
cpu_active_4[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 4.
cpu_active_5[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 5.
cpu_active_6[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 6.
cpu_active_7[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 7.
cpu_active_8[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 8.
cpu_active_9[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 9.
po_reset	*Signal* on page 2-85	Slave	Power on reset.
ppi*x*_in_*n*[8]	*Signal* on page 2-85	Slave	Private peripheral interrupts (ID*x*) of cluster *n*, where *x* is in the range 16-31 and *n* is in the range 0-31. • ID22 is typically the Debug Communications Channel (DCC) interrupt. • ID23 is typically the Performance Counter (PMU) overflow interrupt. • ID24 is typically the Cross Trigger Interface (CTI) interrupt. • ID25 is typically the virtual CPU interface maintenance interrupt. • ID26 is typically the hypervisor timer. • ID27 is typically the virtual timer. • ID29 is typically the Secure physical timer. • ID30 is typically the Non-secure physical timer.
pvbus_m	PVBus	Master	Memory bus out: transactions generated by the IRI.
pvbus_s	PVBus	Slave	Memory bus in: memory-mapped register accesses are accepted on this interface.
redistributor_m[256]	*GICv3Comms* on page 2-84	Master	Input from and output to CPU interface.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-348 Ports (continued)

Name	Protocol	Type	Description
reset	Signal on page 2-85	Slave	Reset.
spi_in[988]	Signal on page 2-85	Slave	Shared peripheral interrupts.
wake_request_0[8]	Signal on page 2-85	Master	Power management outputs of cluster 0.
wake_request_1[8]	Signal on page 2-85	Master	Power management outputs of cluster 1.
wake_request_10[8]	Signal on page 2-85	Master	Power management outputs of cluster 10.
wake_request_11[8]	Signal on page 2-85	Master	Power management outputs of cluster 11.
wake_request_12[8]	Signal on page 2-85	Master	Power management outputs of cluster 12.
wake_request_13[8]	Signal on page 2-85	Master	Power management outputs of cluster 13.
wake_request_14[8]	Signal on page 2-85	Master	Power management outputs of cluster 14.
wake_request_15[8]	Signal on page 2-85	Master	Power management outputs of cluster 15.
wake_request_16[8]	Signal on page 2-85	Master	Power management outputs of cluster 16.
wake_request_17[8]	Signal on page 2-85	Master	Power management outputs of cluster 17.
wake_request_18[8]	Signal on page 2-85	Master	Power management outputs of cluster 18.
wake_request_19[8]	Signal on page 2-85	Master	Power management outputs of cluster 19.
wake_request_2[8]	Signal on page 2-85	Master	Power management outputs of cluster 2.
wake_request_20[8]	Signal on page 2-85	Master	Power management outputs of cluster 20.
wake_request_21[8]	Signal on page 2-85	Master	Power management outputs of cluster 21.
wake_request_22[8]	Signal on page 2-85	Master	Power management outputs of cluster 22.
wake_request_23[8]	Signal on page 2-85	Master	Power management outputs of cluster 23.
wake_request_24[8]	Signal on page 2-85	Master	Power management outputs of cluster 24.
wake_request_25[8]	Signal on page 2-85	Master	Power management outputs of cluster 25.
wake_request_26[8]	Signal on page 2-85	Master	Power management outputs of cluster 26.
wake_request_27[8]	Signal on page 2-85	Master	Power management outputs of cluster 27.
wake_request_28[8]	Signal on page 2-85	Master	Power management outputs of cluster 28.
wake_request_29[8]	Signal on page 2-85	Master	Power management outputs of cluster 29.
wake_request_3[8]	Signal on page 2-85	Master	Power management outputs of cluster 3.
wake_request_30[8]	Signal on page 2-85	Master	Power management outputs of cluster 30.
wake_request_31[8]	Signal on page 2-85	Master	Power management outputs of cluster 31.
wake_request_4[8]	Signal on page 2-85	Master	Power management outputs of cluster 4.
wake_request_5[8]	Signal on page 2-85	Master	Power management outputs of cluster 5.
wake_request_6[8]	Signal on page 2-85	Master	Power management outputs of cluster 6.
wake_request_7[8]	Signal on page 2-85	Master	Power management outputs of cluster 7.
wake_request_8[8]	Signal on page 2-85	Master	Power management outputs of cluster 8.
wake_request_9[8]	Signal on page 2-85	Master	Power management outputs of cluster 9.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-349 Parameters for GIC500

Name	Type	Default value	Description
GICD_ITARGETSR-RAZWI	bool	0x0	If true, the GICD_ITARGETS registers are RAZ/WI
ITS-count	int	0x1	Number of Interrupt Translation Services to be instantiated (0=none)
ITS-device-bits	int	0x10	Number of bits supported for ITS device IDs.
ITS-threaded-command-queue	bool	0x1	Enable execution of ITS commands in a separate thread which is sometimes required for cosimulation
SPI-count	int	0xe0	Number of SPIs that are implemented.
cpus_per_cluster_0	int	0x1	Number of cores within cluster 0
cpus_per_cluster_1	int	0x1	Number of cores within cluster 1
cpus_per_cluster_10	int	0x1	Number of cores within cluster 10
cpus_per_cluster_11	int	0x1	Number of cores within cluster 11
cpus_per_cluster_12	int	0x1	Number of cores within cluster 12
cpus_per_cluster_13	int	0x1	Number of cores within cluster 13
cpus_per_cluster_14	int	0x1	Number of cores within cluster 14
cpus_per_cluster_15	int	0x1	Number of cores within cluster 15
cpus_per_cluster_16	int	0x1	Number of cores within cluster 16
cpus_per_cluster_17	int	0x1	Number of cores within cluster 17
cpus_per_cluster_18	int	0x1	Number of cores within cluster 18
cpus_per_cluster_19	int	0x1	Number of cores within cluster 19
cpus_per_cluster_2	int	0x1	Number of cores within cluster 2
cpus_per_cluster_20	int	0x1	Number of cores within cluster 20
cpus_per_cluster_21	int	0x1	Number of cores within cluster 21
cpus_per_cluster_22	int	0x1	Number of cores within cluster 22
cpus_per_cluster_23	int	0x1	Number of cores within cluster 23
cpus_per_cluster_24	int	0x1	Number of cores within cluster 24
cpus_per_cluster_25	int	0x1	Number of cores within cluster 25
cpus_per_cluster_26	int	0x1	Number of cores within cluster 26
cpus_per_cluster_27	int	0x1	Number of cores within cluster 27
cpus_per_cluster_28	int	0x1	Number of cores within cluster 28
cpus_per_cluster_29	int	0x1	Number of cores within cluster 29
cpus_per_cluster_3	int	0x1	Number of cores within cluster 3
cpus_per_cluster_30	int	0x1	Number of cores within cluster 30
cpus_per_cluster_31	int	0x1	Number of cores within cluster 31
cpus_per_cluster_4	int	0x1	Number of cores within cluster 4
cpus_per_cluster_5	int	0x1	Number of cores within cluster 5
cpus_per_cluster_6	int	0x1	Number of cores within cluster 6

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-349 Parameters for GIC500 (continued)

Name	Type	Default value	Description
cpus_per_cluster_7	int	0x1	Number of cores within cluster 7
cpus_per_cluster_8	int	0x1	Number of cores within cluster 8
cpus_per_cluster_9	int	0x1	Number of cores within cluster 9
delay-ITS-accesses	bool	0x1	Delay accesses from the ITS until GICR_SYNCR is read.
delay-redistributor-accesses	bool	0x1	Delay memory accesses from the redistributor until GICR_SYNCR is read.
enable_protocol_checking	bool	0x0	Enable/disable protocol checking at cpu interface
enabled	bool	0x1	Enable GICv3 functionality; when false the component is inactive.
has-two-security-states	bool	0x1	If true, has two security states
num_clusters	int	0x1	Number of implemented affinity level1 clusters
print-memory-map	bool	0x0	Print memory map to stdout
redistributor-threaded-sync	bool	0x1	Enable execution of redistributor delayed transactions in a separate thread which is sometimes required for cosimulation
reg-base	int	0x2c010000	GIC500 base address
using-generated-memorymap	bool	0x1	Use the generated memorymap for the GIC500 and warn if superfluous parameters are passed
wakeup-on-reset	bool	0x0	Go against specification and start redistributors in woken-up state at reset. This allows software that was written for previous versions of the GICv3 specification to work correctly. This should not be used for production code or when the distributor is used separately from the core fast model.

3.9.20 GIC500_ClusterPorts

GIC500 Component for distribution of interrupts. This model is written in C++.

GIC500_ClusterPorts contains the following CADI targets:

- GIC500

GIC500_ClusterPorts contains the following MTI components:

- *4.76 GICv3Distributor - trace* on page 4-2496
- *4.77 GICv3IRI - trace* on page 4-2504
- *4.78 GICv3InterruptTranslationService - trace* on page 4-2505
- *4.80 GICv3Redistributor - trace* on page 4-2527
- *4.81 GICv3RedistributorInternal - trace* on page 4-2541

Table 3-350 Ports

Name	Protocol	Type	Description
cfgsdisable	*Signal* on page 2-85	Slave	Disable some SPIs signal.
cpu_active_0[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 0.
cpu_active_1[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 1.
cpu_active_10[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 10.
cpu_active_11[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 11.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-350 Ports (continued)

Name	Protocol	Type	Description
cpu_active_12[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 12.
cpu_active_13[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 13.
cpu_active_14[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 14.
cpu_active_15[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 15.
cpu_active_16[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 16.
cpu_active_17[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 17.
cpu_active_18[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 18.
cpu_active_19[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 19.
cpu_active_2[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 2.
cpu_active_20[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 20.
cpu_active_21[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 21.
cpu_active_22[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 22.
cpu_active_23[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 23.
cpu_active_24[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 24.
cpu_active_25[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 25.
cpu_active_26[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 26.
cpu_active_27[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 27.
cpu_active_28[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 28.
cpu_active_29[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 29.
cpu_active_3[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 3.
cpu_active_30[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 30.
cpu_active_31[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 31.
cpu_active_4[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 4.
cpu_active_5[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 5.
cpu_active_6[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 6.
cpu_active_7[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 7.
cpu_active_8[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 8.
cpu_active_9[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 9.
po_reset	*Signal* on page 2-85	Slave	Power on reset.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-350 Ports (continued)

Name	Protocol	Type	Description
ppi*x*_in_*n*[8]	*Signal* on page 2-85	Slave	Private peripheral interrupts (ID*x*) of cluster *n*, where *x* is in the range 16-31 and *n* is in the range 0-31. • ID22 is typically the Debug Communications Channel (DCC) interrupt. • ID23 is typically the Performance Counter (PMU) overflow interrupt. • ID24 is typically the Cross Trigger Interface (CTI) interrupt. • ID25 is typically the virtual CPU interface maintenance interrupt. • ID26 is typically the hypervisor timer. • ID27 is typically the virtual timer. • ID29 is typically the Secure physical timer. • ID30 is typically the Non-secure physical timer.
pvbus_m	PVBus	Master	Memory bus out: transactions generated by the IRI.
pvbus_s	PVBus	Slave	Memory bus in: memory-mapped register accesses are accepted on this interface.
redistributor_0[8]	*GICv3Comms* on page 2-84	Master	Input from and output to CPU interface for cluster 0
redistributor_1[8]	*GICv3Comms* on page 2-84	Master	Input from and output to CPU interface for cluster 1.
redistributor_10[8]	*GICv3Comms* on page 2-84	Master	Input from and output to CPU interface for cluster 10.
redistributor_11[8]	*GICv3Comms* on page 2-84	Master	Input from and output to CPU interface for cluster 11.
redistributor_12[8]	*GICv3Comms* on page 2-84	Master	Input from and output to CPU interface for cluster 12.
redistributor_13[8]	*GICv3Comms* on page 2-84	Master	Input from and output to CPU interface for cluster 13.
redistributor_14[8]	*GICv3Comms* on page 2-84	Master	Input from and output to CPU interface for cluster 14.
redistributor_15[8]	*GICv3Comms* on page 2-84	Master	Input from and output to CPU interface for cluster 15.
redistributor_16[8]	*GICv3Comms* on page 2-84	Master	Input from and output to CPU interface for cluster 16.
redistributor_17[8]	*GICv3Comms* on page 2-84	Master	Input from and output to CPU interface for cluster 17.
redistributor_18[8]	*GICv3Comms* on page 2-84	Master	Input from and output to CPU interface for cluster 18.
redistributor_19[8]	*GICv3Comms* on page 2-84	Master	Input from and output to CPU interface for cluster 19.
redistributor_2[8]	*GICv3Comms* on page 2-84	Master	Input from and output to CPU interface for cluster 2.
redistributor_20[8]	*GICv3Comms* on page 2-84	Master	Input from and output to CPU interface for cluster 20
redistributor_21[8]	*GICv3Comms* on page 2-84	Master	Input from and output to CPU interface for cluster 21.
redistributor_22[8]	*GICv3Comms* on page 2-84	Master	Input from and output to CPU interface for cluster 22.
redistributor_23[8]	*GICv3Comms* on page 2-84	Master	Input from and output to CPU interface for cluster 23
redistributor_24[8]	*GICv3Comms* on page 2-84	Master	Input from and output to CPU interface for cluster 24.
redistributor_25[8]	*GICv3Comms* on page 2-84	Master	Input from and output to CPU interface for cluster 25.
redistributor_26[8]	*GICv3Comms* on page 2-84	Master	Input from and output to CPU interface for cluster 26.
redistributor_27[8]	*GICv3Comms* on page 2-84	Master	Input from and output to CPU interface for cluster 27.
redistributor_28[8]	*GICv3Comms* on page 2-84	Master	Input from and output to CPU interface for cluster 28.
redistributor_29[8]	*GICv3Comms* on page 2-84	Master	Input from and output to CPU interface for cluster 29.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-350 Ports (continued)

Name	Protocol	Type	Description
redistributor_3[8]	GICv3Comms on page 2-84	Master	Input from and output to CPU interface for cluster 3.
redistributor_30[8]	GICv3Comms on page 2-84	Master	Input from and output to CPU interface for cluster 30.
redistributor_31[8]	GICv3Comms on page 2-84	Master	Input from and output to CPU interface for cluster 31.
redistributor_4[8]	GICv3Comms on page 2-84	Master	Input from and output to CPU interface for cluster 4.
redistributor_5[8]	GICv3Comms on page 2-84	Master	Input from and output to CPU interface for cluster 5.
redistributor_6[8]	GICv3Comms on page 2-84	Master	Input from and output to CPU interface for cluster 6.
redistributor_7[8]	GICv3Comms on page 2-84	Master	Input from and output to CPU interface for cluster 7.
redistributor_8[8]	GICv3Comms on page 2-84	Master	Input from and output to CPU interface for cluster 8.
redistributor_9[8]	GICv3Comms on page 2-84	Master	Input from and output to CPU interface for cluster 9.
reset	Signal on page 2-85	Slave	Reset.
spi_in[988]	Signal on page 2-85	Slave	Shared peripheral interrupts.
wake_request_0[8]	Signal on page 2-85	Master	Power management outputs of cluster 0.
wake_request_1[8]	Signal on page 2-85	Master	Power management outputs of cluster 1.
wake_request_10[8]	Signal on page 2-85	Master	Power management outputs of cluster 10.
wake_request_11[8]	Signal on page 2-85	Master	Power management outputs of cluster 11.
wake_request_12[8]	Signal on page 2-85	Master	Power management outputs of cluster 12.
wake_request_13[8]	Signal on page 2-85	Master	Power management outputs of cluster 13.
wake_request_14[8]	Signal on page 2-85	Master	Power management outputs of cluster 14.
wake_request_15[8]	Signal on page 2-85	Master	Power management outputs of cluster 15.
wake_request_16[8]	Signal on page 2-85	Master	Power management outputs of cluster 16.
wake_request_17[8]	Signal on page 2-85	Master	Power management outputs of cluster 17.
wake_request_18[8]	Signal on page 2-85	Master	Power management outputs of cluster 18.
wake_request_19[8]	Signal on page 2-85	Master	Power management outputs of cluster 19.
wake_request_2[8]	Signal on page 2-85	Master	Power management outputs of cluster 2.
wake_request_20[8]	Signal on page 2-85	Master	Power management outputs of cluster 20.
wake_request_21[8]	Signal on page 2-85	Master	Power management outputs of cluster 21.
wake_request_22[8]	Signal on page 2-85	Master	Power management outputs of cluster 22.
wake_request_23[8]	Signal on page 2-85	Master	Power management outputs of cluster 23.
wake_request_24[8]	Signal on page 2-85	Master	Power management outputs of cluster 24.
wake_request_25[8]	Signal on page 2-85	Master	Power management outputs of cluster 25.
wake_request_26[8]	Signal on page 2-85	Master	Power management outputs of cluster 26.
wake_request_27[8]	Signal on page 2-85	Master	Power management outputs of cluster 27.
wake_request_28[8]	Signal on page 2-85	Master	Power management outputs of cluster 28.
wake_request_29[8]	Signal on page 2-85	Master	Power management outputs of cluster 29.
wake_request_3[8]	Signal on page 2-85	Master	Power management outputs of cluster 3.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-350 Ports (continued)

Name	Protocol	Type	Description
wake_request_30[8]	*Signal* on page 2-85	Master	Power management outputs of cluster 30.
wake_request_31[8]	*Signal* on page 2-85	Master	Power management outputs of cluster 31.
wake_request_4[8]	*Signal* on page 2-85	Master	Power management outputs of cluster 4.
wake_request_5[8]	*Signal* on page 2-85	Master	Power management outputs of cluster 5.
wake_request_6[8]	*Signal* on page 2-85	Master	Power management outputs of cluster 6.
wake_request_7[8]	*Signal* on page 2-85	Master	Power management outputs of cluster 7.
wake_request_8[8]	*Signal* on page 2-85	Master	Power management outputs of cluster 8.
wake_request_9[8]	*Signal* on page 2-85	Master	Power management outputs of cluster 9.

Table 3-351 Parameters for GIC500

Name	Type	Default value	Description
GICD_ITARGETSR-RAZWI	bool	0x0	If true, the GICD_ITARGETS registers are RAZ/WI
ITS-count	int	0x1	Number of Interrupt Translation Services to be instantiated (0=none)
ITS-device-bits	int	0x10	Number of bits supported for ITS device IDs.
ITS-threaded-command-queue	bool	0x1	Enable execution of ITS commands in a separate thread which is sometimes required for cosimulation
SPI-count	int	0xe0	Number of SPIs that are implemented.
cpus_per_cluster_0	int	0x1	Number of cores within cluster 0
cpus_per_cluster_1	int	0x1	Number of cores within cluster 1
cpus_per_cluster_10	int	0x1	Number of cores within cluster 10
cpus_per_cluster_11	int	0x1	Number of cores within cluster 11
cpus_per_cluster_12	int	0x1	Number of cores within cluster 12
cpus_per_cluster_13	int	0x1	Number of cores within cluster 13
cpus_per_cluster_14	int	0x1	Number of cores within cluster 14
cpus_per_cluster_15	int	0x1	Number of cores within cluster 15
cpus_per_cluster_16	int	0x1	Number of cores within cluster 16
cpus_per_cluster_17	int	0x1	Number of cores within cluster 17
cpus_per_cluster_18	int	0x1	Number of cores within cluster 18
cpus_per_cluster_19	int	0x1	Number of cores within cluster 19
cpus_per_cluster_2	int	0x1	Number of cores within cluster 2
cpus_per_cluster_20	int	0x1	Number of cores within cluster 20
cpus_per_cluster_21	int	0x1	Number of cores within cluster 21
cpus_per_cluster_22	int	0x1	Number of cores within cluster 22
cpus_per_cluster_23	int	0x1	Number of cores within cluster 23
cpus_per_cluster_24	int	0x1	Number of cores within cluster 24

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-351 Parameters for GIC500 (continued)

Name	Type	Default value	Description
cpus_per_cluster_25	int	0x1	Number of cores within cluster 25
cpus_per_cluster_26	int	0x1	Number of cores within cluster 26
cpus_per_cluster_27	int	0x1	Number of cores within cluster 27
cpus_per_cluster_28	int	0x1	Number of cores within cluster 28
cpus_per_cluster_29	int	0x1	Number of cores within cluster 29
cpus_per_cluster_3	int	0x1	Number of cores within cluster 3
cpus_per_cluster_30	int	0x1	Number of cores within cluster 30
cpus_per_cluster_31	int	0x1	Number of cores within cluster 31
cpus_per_cluster_4	int	0x1	Number of cores within cluster 4
cpus_per_cluster_5	int	0x1	Number of cores within cluster 5
cpus_per_cluster_6	int	0x1	Number of cores within cluster 6
cpus_per_cluster_7	int	0x1	Number of cores within cluster 7
cpus_per_cluster_8	int	0x1	Number of cores within cluster 8
cpus_per_cluster_9	int	0x1	Number of cores within cluster 9
delay-ITS-accesses	bool	0x1	Delay accesses from the ITS until GICR_SYNCR is read.
delay-redistributor-accesses	bool	0x1	Delay memory accesses from the redistributor until GICR_SYNCR is read.
enable_protocol_checking	bool	0x0	Enable/disable protocol checking at cpu interface
enabled	bool	0x1	Enable GICv3 functionality; when false the component is inactive.
has-two-security-states	bool	0x1	If true, has two security states
num_clusters	int	0x1	Number of implemented affinity level1 clusters
print-memory-map	bool	0x0	Print memory map to stdout
redistributor-threaded-sync	bool	0x1	Enable execution of redistributor delayed transactions in a separate thread which is sometimes required for cosimulation
reg-base	int	0x2c010000	GIC500 base address
using-generated-memorymap	bool	0x1	Use the generated memorymap for the GIC500 and warn if superfluous parameters are passed
wakeup-on-reset	bool	0x0	Go against specification and start redistributors in woken-up state at reset. This allows software that was written for previous versions of the GICv3 specification to work correctly. This should not be used for production code or when the distributor is used separately from the core fast model.

3.9.21 GIC500_Filter

GIC500 Component for distribution of interrupts, filtering version for validation. This model is written in C++.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

GIC500_Filter contains the following CADI targets:

- GIC500

GIC500_Filter contains the following MTI components:

- *4.76 GICv3Distributor - trace* on page 4-2496
- *4.77 GICv3IRI - trace* on page 4-2504
- *4.78 GICv3InterruptTranslationService - trace* on page 4-2505
- *4.80 GICv3Redistributor - trace* on page 4-2527
- *4.81 GICv3RedistributorInternal - trace* on page 4-2541

About GIC500_Filter

This is a single-component implementation of the GICv3 architecture with support for 256 cores. You can configure the model to support a maximum of 32 clusters with 8 cores per cluster. Use it with an Armv8-A core to deliver interrupts. It supports a single Interrupt Translation Service for message-based interrupts. It supports the architectural features, but does not support the IMPLEMENTATION DEFINED features.

Note

- Set the `FASTSIM_GIC_MEMORY_MAP` environment variable to 1 to print to `stderr` the memory map of any GICv3 or later models that are included in the platform being run.
- For writes to `GITS_TRANSLATE64R`, the *MSIRewriter* on page 3-104 component can be used to create the correct data transactions based on the device ID.

MasterID, ExtendedID, and UserFlags

This table shows how this model encodes information in the `MasterID`, `ExtendedID`, and `UserFlags` bus attributes:

Table 3-352 pvbus_s port

PVBus attribute	Bits used	Property encoded	Notes
`MasterID`	Bits[31:0]	Master ID that invoked the register access	-
`ExtendedID`	-	-	Not used.
`UserFlags`	-	-	Not used.

Note

The `pvbus_m` port does not use `MasterID`, `ExtendedID`, or `UserFlags`.

Table 3-353 Ports

Name	Protocol	Type	Description
`cfgsdisable`	*Signal* on page 2-85	Slave	Disable some SPIs signal.
`cpu_active_0[8]`	*Signal* on page 2-85	Slave	cpu_active pins of cluster 0.
`cpu_active_1[8]`	*Signal* on page 2-85	Slave	cpu_active pins of cluster 1.
`cpu_active_10[8]`	*Signal* on page 2-85	Slave	cpu_active pins of cluster 10.
`cpu_active_11[8]`	*Signal* on page 2-85	Slave	cpu_active pins of cluster 11.
`cpu_active_12[8]`	*Signal* on page 2-85	Slave	cpu_active pins of cluster 12.
`cpu_active_13[8]`	*Signal* on page 2-85	Slave	cpu_active pins of cluster 13.
`cpu_active_14[8]`	*Signal* on page 2-85	Slave	cpu_active pins of cluster 14.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-353 Ports (continued)

Name	Protocol	Type	Description
cpu_active_15[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 15.
cpu_active_16[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 16.
cpu_active_17[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 17.
cpu_active_18[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 18.
cpu_active_19[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 19.
cpu_active_2[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 2.
cpu_active_20[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 20.
cpu_active_21[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 21.
cpu_active_22[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 22.
cpu_active_23[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 23.
cpu_active_24[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 24.
cpu_active_25[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 25.
cpu_active_26[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 26.
cpu_active_27[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 27.
cpu_active_28[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 28.
cpu_active_29[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 29.
cpu_active_3[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 3.
cpu_active_30[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 30.
cpu_active_31[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 31.
cpu_active_4[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 4.
cpu_active_5[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 5.
cpu_active_6[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 6.
cpu_active_7[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 7.
cpu_active_8[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 8.
cpu_active_9[8]	*Signal* on page 2-85	Slave	cpu_active pins of cluster 9.
po_reset	*Signal* on page 2-85	Slave	Power on reset.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-353 Ports (continued)

Name	Protocol	Type	Description
ppi*x*_in_*n*[8]	*Signal* on page 2-85	Slave	Private peripheral interrupts (ID*x*) of cluster *n*, where *x* is in the range 16-31 and *n* is in the range 0-31. • ID22 is typically the Debug Communications Channel (DCC) interrupt. • ID23 is typically the Performance Counter (PMU) overflow interrupt. • ID24 is typically the Cross Trigger Interface (CTI) interrupt. • ID25 is typically the virtual CPU interface maintenance interrupt. • ID26 is typically the hypervisor timer. • ID27 is typically the virtual timer. • ID29 is typically the Secure physical timer. • ID30 is typically the Non-secure physical timer.
pvbus_filtermiss_m	PVBus	Master	passthrough for transactions not targetting one of the pages associated with the IRI.
pvbus_m	PVBus	Master	Memory bus out: transactions generated by the IRI.
pvbus_s	PVBus	Slave	Memory bus in: memory-mapped register accesses are accepted on this interface.
redistributor_m[256]	*GICv3Comms* on page 2-84	Master	Input from and output to CPU interface.
reset	*Signal* on page 2-85	Slave	Reset.
spi_in[988]	*Signal* on page 2-85	Slave	Shared peripheral interrupts.
wake_request_0[8]	*Signal* on page 2-85	Master	Power management outputs of cluster 0.
wake_request_1[8]	*Signal* on page 2-85	Master	Power management outputs of cluster 1.
wake_request_10[8]	*Signal* on page 2-85	Master	Power management outputs of cluster 10.
wake_request_11[8]	*Signal* on page 2-85	Master	Power management outputs of cluster 11.
wake_request_12[8]	*Signal* on page 2-85	Master	Power management outputs of cluster 12.
wake_request_13[8]	*Signal* on page 2-85	Master	Power management outputs of cluster 13.
wake_request_14[8]	*Signal* on page 2-85	Master	Power management outputs of cluster 14.
wake_request_15[8]	*Signal* on page 2-85	Master	Power management outputs of cluster 15.
wake_request_16[8]	*Signal* on page 2-85	Master	Power management outputs of cluster 16.
wake_request_17[8]	*Signal* on page 2-85	Master	Power management outputs of cluster 17.
wake_request_18[8]	*Signal* on page 2-85	Master	Power management outputs of cluster 18.
wake_request_19[8]	*Signal* on page 2-85	Master	Power management outputs of cluster 19.
wake_request_2[8]	*Signal* on page 2-85	Master	Power management outputs of cluster 2.
wake_request_20[8]	*Signal* on page 2-85	Master	Power management outputs of cluster 20.
wake_request_21[8]	*Signal* on page 2-85	Master	Power management outputs of cluster 21.
wake_request_22[8]	*Signal* on page 2-85	Master	Power management outputs of cluster 22.
wake_request_23[8]	*Signal* on page 2-85	Master	Power management outputs of cluster 23.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-353 Ports (continued)

Name	Protocol	Type	Description
wake_request_24[8]	*Signal* on page 2-85	Master	Power management outputs of cluster 24.
wake_request_25[8]	*Signal* on page 2-85	Master	Power management outputs of cluster 25.
wake_request_26[8]	*Signal* on page 2-85	Master	Power management outputs of cluster 26.
wake_request_27[8]	*Signal* on page 2-85	Master	Power management outputs of cluster 27.
wake_request_28[8]	*Signal* on page 2-85	Master	Power management outputs of cluster 28.
wake_request_29[8]	*Signal* on page 2-85	Master	Power management outputs of cluster 29.
wake_request_3[8]	*Signal* on page 2-85	Master	Power management outputs of cluster 3.
wake_request_30[8]	*Signal* on page 2-85	Master	Power management outputs of cluster 30.
wake_request_31[8]	*Signal* on page 2-85	Master	Power management outputs of cluster 31.
wake_request_4[8]	*Signal* on page 2-85	Master	Power management outputs of cluster 4.
wake_request_5[8]	*Signal* on page 2-85	Master	Power management outputs of cluster 5.
wake_request_6[8]	*Signal* on page 2-85	Master	Power management outputs of cluster 6.
wake_request_7[8]	*Signal* on page 2-85	Master	Power management outputs of cluster 7.
wake_request_8[8]	*Signal* on page 2-85	Master	Power management outputs of cluster 8.
wake_request_9[8]	*Signal* on page 2-85	Master	Power management outputs of cluster 9.

Table 3-354 Parameters for GIC500

Name	Type	Default value	Description
GICD_ITARGETSR-RAZWI	bool	0x0	If true, the GICD_ITARGETS registers are RAZ/WI
ITS-count	int	0x1	Number of Interrupt Translation Services to be instantiated (0=none)
ITS-device-bits	int	0x10	Number of bits supported for ITS device IDs.
ITS-threaded-command-queue	bool	0x1	Enable execution of ITS commands in a separate thread which is sometimes required for cosimulation
SPI-count	int	0xe0	Number of SPIs that are implemented.
cpus_per_cluster_0	int	0x1	Number of cores within cluster 0
cpus_per_cluster_1	int	0x1	Number of cores within cluster 1
cpus_per_cluster_10	int	0x1	Number of cores within cluster 10
cpus_per_cluster_11	int	0x1	Number of cores within cluster 11
cpus_per_cluster_12	int	0x1	Number of cores within cluster 12
cpus_per_cluster_13	int	0x1	Number of cores within cluster 13
cpus_per_cluster_14	int	0x1	Number of cores within cluster 14
cpus_per_cluster_15	int	0x1	Number of cores within cluster 15
cpus_per_cluster_16	int	0x1	Number of cores within cluster 16
cpus_per_cluster_17	int	0x1	Number of cores within cluster 17
cpus_per_cluster_18	int	0x1	Number of cores within cluster 18

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-354 Parameters for GIC500 (continued)

Name	Type	Default value	Description
cpus_per_cluster_19	int	0x1	Number of cores within cluster 19
cpus_per_cluster_2	int	0x1	Number of cores within cluster 2
cpus_per_cluster_20	int	0x1	Number of cores within cluster 20
cpus_per_cluster_21	int	0x1	Number of cores within cluster 21
cpus_per_cluster_22	int	0x1	Number of cores within cluster 22
cpus_per_cluster_23	int	0x1	Number of cores within cluster 23
cpus_per_cluster_24	int	0x1	Number of cores within cluster 24
cpus_per_cluster_25	int	0x1	Number of cores within cluster 25
cpus_per_cluster_26	int	0x1	Number of cores within cluster 26
cpus_per_cluster_27	int	0x1	Number of cores within cluster 27
cpus_per_cluster_28	int	0x1	Number of cores within cluster 28
cpus_per_cluster_29	int	0x1	Number of cores within cluster 29
cpus_per_cluster_3	int	0x1	Number of cores within cluster 3
cpus_per_cluster_30	int	0x1	Number of cores within cluster 30
cpus_per_cluster_31	int	0x1	Number of cores within cluster 31
cpus_per_cluster_4	int	0x1	Number of cores within cluster 4
cpus_per_cluster_5	int	0x1	Number of cores within cluster 5
cpus_per_cluster_6	int	0x1	Number of cores within cluster 6
cpus_per_cluster_7	int	0x1	Number of cores within cluster 7
cpus_per_cluster_8	int	0x1	Number of cores within cluster 8
cpus_per_cluster_9	int	0x1	Number of cores within cluster 9
delay-ITS-accesses	bool	0x1	Delay accesses from the ITS until GICR_SYNCR is read.
delay-redistributor-accesses	bool	0x1	Delay memory accesses from the redistributor until GICR_SYNCR is read.
enable_protocol_checking	bool	0x0	Enable/disable protocol checking at cpu interface
enabled	bool	0x1	Enable GICv3 functionality; when false the component is inactive.
has-two-security-states	bool	0x1	If true, has two security states
num_clusters	int	0x1	Number of implemented affinity level1 clusters
print-memory-map	bool	0x0	Print memory map to stdout
redistributor-threaded-sync	bool	0x1	Enable execution of redistributor delayed transactions in a separate thread which is sometimes required for cosimulation
reg-base	int	0x2c010000	GIC500 base address

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-354 Parameters for GIC500 (continued)

Name	Type	Default value	Description
`using-generated-memorymap`	`bool`	`0x1`	Use the generated memorymap for the GIC500 and warn if superfluous parameters are passed
`wakeup-on-reset`	`bool`	`0x0`	Go against specification and start redistributors in woken-up state at reset. This allows software that was written for previous versions of the GICv3 specification to work correctly. This should not be used for production code or when the distributor is used separately from the core fast model.

3.9.22 GIC600

GIC-600 IRI implementation: Single chip validation/filter component variant limited to 265 PE. This model is written in C++.

GIC600 contains the following CADI targets:

- GIC600

GIC600 contains the following MTI components:

- *4.76 GICv3Distributor - trace* on page 4-2496
- *4.77 GICv3IRI - trace* on page 4-2504
- *4.78 GICv3InterruptTranslationService - trace* on page 4-2505
- *4.79 GICv3ProtocolChecker - trace* on page 4-2526
- *4.80 GICv3Redistributor - trace* on page 4-2527
- *4.81 GICv3RedistributorInternal - trace* on page 4-2541

About GIC600 and GIC600_Filter

GIC600 and GIC600_Filter are minimal models of an Arm GIC-600 Generic Interrupt Controller, suitable for single-chip or multichip systems, where multichip operation has been tested, with the following exceptions:

- LPI/LPI command is implemented but untested.
- Power operation is unimplemented.

They provide a simple configuration interface that allows designers to introduce GIC600-like functionality to their systems, while only implementing the architectural behavior, as defined by the GICv3 architecture.

All implementation-specific registers and functionality are implemented.

As with the other GIC components, there are two variants of the model with slightly different memory interfaces. Both GIC600 and GIC600_Filter have a `pvbus_s` port for register accesses and a `pvbus_m` port for the LPI-related traffic from redistributors and the ITS. The `pvbus_m` port is also used to compose multichip operation.

In addition, the GIC600_Filter variant has a `pvbus_filtermiss_m` port, to which any transactions on the `pvbus_s` port and not directed to a 4K page used by the GIC are forwarded unmodified. Such transactions are terminated in the component when using the GIC600 variant.

It is recommended to use the GIC600 variant in most cases.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Also, the RAS register is implemented and various RAS errors are reported through status registers. Not all the RAS errors are available because Fast Models does not model errors that can cause ECC errors. Fast Models supports error record 0 and error record 13+.

Note

- Set the `FASTSIM_GIC_MEMORY_MAP` environment variable to 1 to print to `stderr` the memory map of any GICv3 or later models that are included in the platform being run.
- For writes to `GITS_TRANSLATE64R`, the *MSIRewriter* on page 3-104 component can be used to create the correct data transactions based on the device ID.

MasterID, ExtendedID, and UserFlags

This table shows how this model encodes information in the `MasterID`, `ExtendedID`, and `UserFlags` bus attributes:

Table 3-355 pvbus_s port

PVBus attribute	Bits used	Property encoded	Notes
`MasterID`	Bits[31:0]	Master ID that invoked the register access	-
`ExtendedID`	-	-	Not used.
`UserFlags`	-	-	Not used.

Note

The `pvbus_m` port does not use `MasterID`, `ExtendedID`, or `UserFlags`.

Table 3-356 Ports

Name	Protocol	Type	Description
`chip_id`	*Value* on page 2-85	Slave	chip_id port which is valid from GIC600 r1p2. Writing to this port for prior GIC600 version will be ignored.
`cpu_active_s[256]`	*Signal* on page 2-85	Slave	CPUActive pins.
`po_reset`	*Signal* on page 2-85	Slave	Resets.
`ppi_in_n[16]`	*Signal* on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu *n*, where *n* is in the range 0-255.
`pvbus_m`	`PVBus`	Master	Memory bus out: transactions generated by the IRI.
`pvbus_s`	`PVBus`	Slave	Memory bus in: memory-mapped register accesses are accepted on this interface.
`redistributor_m[256]`	*GICv3Comms* on page 2-84	Master	Input from and output to CPU interface.
`reset`	*Signal* on page 2-85	Slave	Resets.
`spi_in[960]`	*Signal* on page 2-85	Slave	Shared peripheral interrupts.
`wake_request[256]`	*Signal* on page 2-85	Master	Power management outputs.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-357 Parameters for GIC600

Name	Type	Default value	Description
ARE-fixed-to-one	bool	0x1	GICv2 compatibility is not supported and GICD_CTLR.ARE_* is always one
CPU-affinities	string	""	A comma separated list of dotted quads containing the affinities of all PEs connected to this IRI. If CPU-affinities-file is specified, this parameter is ignored.
CPU-affinities-file	string	""	A file containing comma separated list of dotted quads containing the affinities of all PEs connected to this IRI. If this parameter is specified, CPU-affinities parameter will be ignored even when it is given.
DS-behaviour	int	0x2	GICD_CTLR.DS field behaviour: 0:RAZ/WI, 1:RAO/WI, 2:RW
IIDR	int	0x0	GICD_IIDR and GICR_IIDR value. When the parameter is not given or is set to zero, the GIC600 model is configured to have the latest revision.
ITS-ID-bits	int	0x10	Number of interrupt bits supported by ITS.
ITS-collection-ID-bits	int	0x8	Number of collection bits supported by ITS (optional parameter, 0 => 16bits support and GITS_TYPER.CIL=0
ITS-count	int	0x1	Number of Interrupt Translation Services to be instantiated (0=none)
ITS-device-bits	int	0x10	Number of bits supported for ITS device IDs.
PPI-count	int	0x10	Selects the number of PPI available for each PE as 8(id22-27,29,30), 12(id 20-31) or 16(id 16-31)
RAS-CFI-support	bool	0x0	If true, fault handling interrupt for corrected errors is supported. Not supported otherwise.
RAS-FI-support	bool	0x0	If true, fault handling interrupt is supported. Not supported otherwise.
RAS-UE-support	bool	0x0	If true, In-band uncorrected error reporing is supported. Not supported otherwise.
RAS-UI-support	bool	0x0	If true, error recovery interrupt for uncorrected errors is supported. Not supported otherwise.
SPI-blocks	int	0x1e	Number of SPI blocks supported by the IRI, each block contains 32 SPIs
affinity-width	string	"0.8.8.8"	A dotted quad indicating the bitwidth of fields at each affinity level
chip-id	int	0x0	Chip ID when multichip operation is enabled.
chip-select-affinity-level	int	0x3	Affinity level 2 or 3 can be used for chip select, affinity width is forced to 0 for the level and above as this model supports one chip only
direct-lpi-support	bool	0x0	Enable support for LPI operations through GICR registers
enabled	bool	0x1	Enable GICv3 functionality; when false the component is inactive.
gict-allow-ns-reset	bool	0x1	If true, non-secure read/write access to GICT register is allowed at reset. Not allowed otherwise. This emulates gict_allow_ns tie-off signal.
max-pe-on-chip	int	0x4	Maximum number of cores on any single chip. This will be used to identify the target chip and core when multichip operation is enabled.
multichip-threaded-dgi	bool	0x1	Enable sending multichip DGI messages in a separate thread. This parameter is valid only if enable-multichip-operation is set.
print-memory-map	bool	0x0	Print memory map to stdout

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-357 Parameters for GIC600 (continued)

Name	Type	Default value	Description
`redistributor-group`	`string`	`""`	Redistributor grouping information with affinity as JSON : {"0": ["0.0.0.0","0.0.0.1"],"1":["0.0.1.0","0.0.1.1"]} where RD with quad 0.0.0.0 and 0.0.0.1 belongs to RD group 0. All the RDs belong to one group when this parameter is not given.
`redistributor-group-file`	`string`	`""`	File path to redistributor grouping information with affinity as JSON. The file uses the same format as "redistributor-group" parameter.
`redistributor-power-managed-by-pwrr`	`bool`	`0x1`	GIC600 dedistributor power management is done by updating GICR_PWRR register
`reg-base`	`int`	`0x2c010000`	GIC-600 base address

3.9.23 GIC600AE

GIC-600AE IRI implementation: Single chip validation/filter component variant limited to 265 PE. This model is written in C++.

GIC600AE contains the following CADI targets:

- GIC600AE

GIC600AE contains the following MTI components:

- *4.76 GICv3Distributor - trace* on page 4-2496
- *4.77 GICv3IRI - trace* on page 4-2504
- *4.78 GICv3InterruptTranslationService - trace* on page 4-2505
- *4.79 GICv3ProtocolChecker - trace* on page 4-2526
- *4.80 GICv3Redistributor - trace* on page 4-2527
- *4.81 GICv3RedistributorInternal - trace* on page 4-2541

About GIC600AE and GIC600AE_Filter

GIC600AE and GIC600AE_Filter are minimal models of an Arm GIC-600AE Generic Interrupt Controller, suitable for single-chip or multichip systems, where multichip operation has been tested, with the following exceptions:

- LPI/LPI command is implemented but untested.
- Power operation is unimplemented.

They provide a simple configuration interface that allows designers to introduce GIC600AE-like functionality to their systems, while only implementing the architectural behavior, as defined by the GICv3 architecture.

All implementation-specific registers and functionality are implemented.

As with the other GIC components, there are two variants of the model with slightly different memory interfaces. Both GIC600AE and GIC600AE_Filter have a `pvbus_s` port for register accesses and a `pvbus_m` port for the LPI-related traffic from redistributors and the ITS. The `pvbus_m` port is also used to compose multichip operation.

In addition, the GIC600AE_Filter variant has a `pvbus_filtermiss_m` port, to which any transactions on the `pvbus_s` port and not directed to a 4K page used by the GIC are forwarded unmodified. Such transactions are terminated in the component when using the GIC600AE variant.

It is recommended to use the GIC600AE variant in most cases.

Also, the RAS register is implemented and various RAS errors are reported through status registers. Not all the RAS errors are available because Fast Models does not model errors that can cause ECC errors.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Fast Models supports error record 0 and error record 13+. FMU registers are also implemented and are accessible by software.

Note

- Set the `FASTSIM_GIC_MEMORY_MAP` environment variable to 1 to print to `stderr` the memory map of any GICv3 or later models that are included in the platform being run.
- For writes to `GITS_TRANSLATE64R`, the *MSIRewriter* on page 3-104 component can be used to create the correct data transactions based on the device ID.

MasterID, ExtendedID, and UserFlags

This table shows how this model encodes information in the `MasterID`, `ExtendedID`, and `UserFlags` bus attributes:

Table 3-358 pvbus_s port

PVBus attribute	Bits used	Property encoded	Notes
`MasterID`	Bits[31:0]	Master ID that invoked the register access	-
`ExtendedID`	-	-	Not used.
`UserFlags`	-	-	Not used.

Note

The `pvbus_m` port does not use `MasterID`, `ExtendedID`, or `UserFlags`.

Table 3-359 Ports

Name	Protocol	Type	Description
`apb_bus`	`PVBus`	Slave	APB4 port to FMU, which will be connected from safety island for FuSa
`chip_id`	*Value* on page 2-85	Slave	chip_id port used for multichip operation
`cpu_active_s[256]`	*Signal* on page 2-85	Slave	CPUActive pins.
`fmu_error_int`	*Signal* on page 2-85	Master	FuSa FMU error interrupt signal
`fmu_fault_int`	*Signal* on page 2-85	Master	FuSa FMU fault interrupt signal
`po_reset`	*Signal* on page 2-85	Slave	Resets. This is used as a dbg_reset in TRM, and also be used for cold reset for PMU and FMU.
`ppi_in_0[16]`	*Signal* on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 0.
`ppi_in_1[16]`	*Signal* on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 1.
`ppi_in_10[16]`	*Signal* on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 10.
`ppi_in_100[16]`	*Signal* on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 100.
`ppi_in_101[16]`	*Signal* on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 101.
`ppi_in_102[16]`	*Signal* on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 102.
`ppi_in_103[16]`	*Signal* on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 103.
`ppi_in_104[16]`	*Signal* on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 104.
`ppi_in_105[16]`	*Signal* on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 105.
`ppi_in_106[16]`	*Signal* on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 106.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-359 Ports (continued)

Name	Protocol	Type	Description
ppi_in_107[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 107.
ppi_in_108[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 108.
ppi_in_109[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 109.
ppi_in_11[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 11.
ppi_in_110[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 110.
ppi_in_111[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 111.
ppi_in_112[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 112.
ppi_in_113[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 113.
ppi_in_114[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 114.
ppi_in_115[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 115.
ppi_in_116[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 116.
ppi_in_117[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 117.
ppi_in_118[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 118.
ppi_in_119[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 119.
ppi_in_12[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 12.
ppi_in_120[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 120.
ppi_in_121[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 121.
ppi_in_122[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 122.
ppi_in_123[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 123.
ppi_in_124[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 124.
ppi_in_125[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 125.
ppi_in_126[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 126.
ppi_in_127[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 127.
ppi_in_128[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 128.
ppi_in_129[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 129.
ppi_in_13[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 13.
ppi_in_130[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 130.
ppi_in_131[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 131.
ppi_in_132[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 132.
ppi_in_133[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 133.
ppi_in_134[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 134.
ppi_in_135[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 135.
ppi_in_136[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 136.
ppi_in_137[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 137.
ppi_in_138[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 138.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-359 Ports (continued)

Name	Protocol	Type	Description
ppi_in_139[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 139.
ppi_in_14[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 14.
ppi_in_140[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 140.
ppi_in_141[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 141.
ppi_in_142[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 142.
ppi_in_143[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 143.
ppi_in_144[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 144.
ppi_in_145[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 145.
ppi_in_146[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 146.
ppi_in_147[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 147.
ppi_in_148[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 148.
ppi_in_149[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 149.
ppi_in_15[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 15.
ppi_in_150[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 150.
ppi_in_151[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 151.
ppi_in_152[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 152.
ppi_in_153[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 153.
ppi_in_154[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 154.
ppi_in_155[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 155.
ppi_in_156[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 156.
ppi_in_157[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 157.
ppi_in_158[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 158.
ppi_in_159[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 159.
ppi_in_16[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 16.
ppi_in_160[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 160.
ppi_in_161[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 161.
ppi_in_162[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 162.
ppi_in_163[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 163.
ppi_in_164[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 164.
ppi_in_165[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 165.
ppi_in_166[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 166.
ppi_in_167[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 167.
ppi_in_168[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 168.
ppi_in_169[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 169.
ppi_in_17[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 17.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-359 Ports (continued)

Name	Protocol	Type	Description
ppi_in_170[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 170.
ppi_in_171[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 171.
ppi_in_172[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 172.
ppi_in_173[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 173.
ppi_in_174[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 174.
ppi_in_175[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 175.
ppi_in_176[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 176.
ppi_in_177[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 177.
ppi_in_178[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 178.
ppi_in_179[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 179.
ppi_in_18[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 18.
ppi_in_180[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 180.
ppi_in_181[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 181.
ppi_in_182[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 182.
ppi_in_183[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 183.
ppi_in_184[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 184.
ppi_in_185[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 185.
ppi_in_186[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 186.
ppi_in_187[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 187.
ppi_in_188[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 188.
ppi_in_189[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 189.
ppi_in_19[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 19.
ppi_in_190[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 190.
ppi_in_191[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 191.
ppi_in_192[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 192.
ppi_in_193[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 193.
ppi_in_194[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 194.
ppi_in_195[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 195.
ppi_in_196[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 196.
ppi_in_197[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 197.
ppi_in_198[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 198.
ppi_in_199[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 199.
ppi_in_2[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 2.
ppi_in_20[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 20.
ppi_in_200[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 200.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-359 Ports (continued)

Name	Protocol	Type	Description
ppi_in_201[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 201.
ppi_in_202[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 202.
ppi_in_203[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 203.
ppi_in_204[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 204.
ppi_in_205[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 205.
ppi_in_206[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 206.
ppi_in_207[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 207.
ppi_in_208[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 208.
ppi_in_209[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 209.
ppi_in_21[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 21.
ppi_in_210[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 210.
ppi_in_211[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 211.
ppi_in_212[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 212.
ppi_in_213[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 213.
ppi_in_214[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 214.
ppi_in_215[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 215.
ppi_in_216[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 216.
ppi_in_217[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 217.
ppi_in_218[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 218.
ppi_in_219[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 219.
ppi_in_22[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 22.
ppi_in_220[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 220.
ppi_in_221[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 221.
ppi_in_222[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 222.
ppi_in_223[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 223.
ppi_in_224[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 224.
ppi_in_225[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 225.
ppi_in_226[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 226.
ppi_in_227[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 227.
ppi_in_228[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 228.
ppi_in_229[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 229.
ppi_in_23[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 23.
ppi_in_230[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 230.
ppi_in_231[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 231.
ppi_in_232[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 232.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-359 Ports (continued)

Name	Protocol	Type	Description
ppi_in_233[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 233.
ppi_in_234[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 234.
ppi_in_235[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 235.
ppi_in_236[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 236.
ppi_in_237[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 237.
ppi_in_238[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 238.
ppi_in_239[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 239.
ppi_in_24[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 24.
ppi_in_240[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 240.
ppi_in_241[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 241.
ppi_in_242[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 242.
ppi_in_243[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 243.
ppi_in_244[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 244.
ppi_in_245[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 245.
ppi_in_246[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 246.
ppi_in_247[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 247.
ppi_in_248[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 248.
ppi_in_249[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 249.
ppi_in_25[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 25.
ppi_in_250[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 250.
ppi_in_251[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 251.
ppi_in_252[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 252.
ppi_in_253[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 253.
ppi_in_254[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 254.
ppi_in_255[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 255.
ppi_in_26[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 26.
ppi_in_27[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 27.
ppi_in_28[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 28.
ppi_in_29[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 29.
ppi_in_3[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 3.
ppi_in_30[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 30.
ppi_in_31[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 31.
ppi_in_32[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 32.
ppi_in_33[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 33.
ppi_in_34[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 34.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-359 Ports (continued)

Name	Protocol	Type	Description
ppi_in_35[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 35.
ppi_in_36[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 36.
ppi_in_37[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 37.
ppi_in_38[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 38.
ppi_in_39[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 39.
ppi_in_4[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 4.
ppi_in_40[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 40.
ppi_in_41[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 41.
ppi_in_42[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 42.
ppi_in_43[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 43.
ppi_in_44[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 44.
ppi_in_45[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 45.
ppi_in_46[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 46.
ppi_in_47[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 47.
ppi_in_48[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 48.
ppi_in_49[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 49.
ppi_in_5[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 5.
ppi_in_50[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 50.
ppi_in_51[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 51.
ppi_in_52[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 52.
ppi_in_53[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 53.
ppi_in_54[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 54.
ppi_in_55[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 55.
ppi_in_56[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 56.
ppi_in_57[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 57.
ppi_in_58[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 58.
ppi_in_59[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 59.
ppi_in_6[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 6.
ppi_in_60[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 60.
ppi_in_61[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 61.
ppi_in_62[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 62.
ppi_in_63[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 63.
ppi_in_64[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 64.
ppi_in_65[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 65.
ppi_in_66[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 66.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-359 Ports (continued)

Name	Protocol	Type	Description
ppi_in_67[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 67.
ppi_in_68[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 68.
ppi_in_69[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 69.
ppi_in_7[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 7.
ppi_in_70[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 70.
ppi_in_71[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 71.
ppi_in_72[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 72.
ppi_in_73[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 73.
ppi_in_74[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 74.
ppi_in_75[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 75.
ppi_in_76[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 76.
ppi_in_77[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 77.
ppi_in_78[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 78.
ppi_in_79[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 79.
ppi_in_8[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 8.
ppi_in_80[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 80.
ppi_in_81[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 81.
ppi_in_82[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 82.
ppi_in_83[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 83.
ppi_in_84[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 84.
ppi_in_85[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 85.
ppi_in_86[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 86.
ppi_in_87[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 87.
ppi_in_88[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 88.
ppi_in_89[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 89.
ppi_in_9[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 9.
ppi_in_90[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 90.
ppi_in_91[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 91.
ppi_in_92[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 92.
ppi_in_93[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 93.
ppi_in_94[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 94.
ppi_in_95[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 95.
ppi_in_96[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 96.
ppi_in_97[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 97.
ppi_in_98[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 98.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-359 Ports (continued)

Name	Protocol	Type	Description
ppi_in_99[16]	*Signal* on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 99.
pvbus_m	PVBus	Master	Memory bus out: transactions generated by the IRI.
pvbus_s	PVBus	Slave	Memory bus in: memory-mapped register accesses are accepted on this interface.
redistributor_m[256]	*GICv3Comms* on page 2-84	Master	Input from and output to CPU interface.
reset	*Signal* on page 2-85	Slave	Resets.
spi_in[960]	*Signal* on page 2-85	Slave	Shared peripheral interrupts.
wake_request[256]	*Signal* on page 2-85	Master	Power management outputs.

Table 3-360 Parameters for GIC600AE

Name	Type	Default value	Description
ARE-fixed-to-one	bool	0x1	GICv2 compatibility is not supported and GICD_CTLR.ARE_* is always one
CPU-affinities	string	""	A comma separated list of dotted quads containing the affinities of all PEs connected to this IRI. If CPU-affinities-file is specified, this parameter is ignored.
CPU-affinities-file	string	""	A file containing comma separated list of dotted quads containing the affinities of all PEs connected to this IRI. If this parameter is specified, CPU-affinities parameter will be ignored even when it is given.
DS-behaviour	int	0x2	GICD_CTLR.DS field behaviour: 0:RAZ/WI, 1:RAO/WI, 2:RW
IIDR	int	0x0	GICD_IIDR and GICR_IIDR value. When the parameter is not given or is set to zero, the GIC600 model is configured to have the latest revision.
ITS-ID-bits	int	0x10	Number of interrupt bits supported by ITS.
ITS-collection-ID-bits	int	0x8	Number of collection bits supported by ITS (optional parameter, 0 => 16bits support and GITS_TYPER.CIL=0
ITS-count	int	0x1	Number of Interrupt Translation Services to be instantiated (0=none)
ITS-device-bits	int	0x10	Number of bits supported for ITS device IDs.
PPI-count	int	0x10	Selects the number of PPI available for each PE as 8(id22-27,29,30), 12(id 20-31) or 16(id 16-31)
RAS-CFI-support	bool	0x0	If true, fault handling interrupt for corrected errors is supported. Not supported otherwise.
RAS-FI-support	bool	0x0	If true, fault handling interrupt is supported. Not supported otherwise.
RAS-UE-support	bool	0x0	If true, In-band uncorrected error reporing is supported. Not supported otherwise.
RAS-UI-support	bool	0x0	If true, error recovery interrupt for uncorrected errors is supported. Not supported otherwise.
SPI-blocks	int	0x1e	Number of SPI blocks supported by the IRI, each block contains 32 SPIs
affinity-width	string	"0.8.8.8"	A dotted quad indicating the bitwidth of fields at each affinity level

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-360 Parameters for GIC600AE (continued)

Name	Type	Default value	Description
chip-id	int	0x0	Chip ID when multichip operation is enabled.
chip-select-affinity-level	int	0x3	Affinity level 2 or 3 can be used for chip select, affinity width is forced to 0 for the level and above as this model supports one chip only
direct-lpi-support	bool	0x0	Enable support for LPI operations through GICR registers
enabled	bool	0x1	Enable GICv3 functionality; when false the component is inactive.
gict-allow-ns-reset	bool	0x1	If true, non-secure read/write access to GICT register is allowed at reset. Not allowed otherwise. This emulates gict_allow_ns tie-off signal.
max-pe-on-chip	int	0x4	Maximum number of cores on any single chip. This will be used to identify the target chip and core when multichip operation is enabled.
multichip-threaded-dgi	bool	0x1	Enable sending multichip DGI messages in a separate thread. This parameter is valid only if enable-multichip-operation is set.
print-memory-map	bool	0x0	Print memory map to stdout
redistributor-group	string	""	Redistributor grouping information with affinity as JSON : {"0": ["0.0.0.0","0.0.0.1"],"1":["0.0.1.0","0.0.1.1"]} where RD with quad 0.0.0.0 and 0.0.0.1 belongs to RD group 0. All the RDs belong to one group when this parameter is not given.
redistributor-group-file	string	""	File path to redistributor grouping information with affinity as JSON. The file uses the same format as "redistributor-group" parameter.
redistributor-power-managed-by-pwrr	bool	0x1	GIC600 dedistributor power management is done by updating GICR_PWRR register
reg-base	int	0x2c010000	GIC-600 base address

3.9.24 GIC600AE_Filter

GIC-600AE IRI implementation: Single chip validation/filter component variant limited to 265 PE. This model is written in C++.

GIC600AE_Filter contains the following CADI targets:

- GIC600AE

GIC600AE_Filter contains the following MTI components:

- *4.76 GICv3Distributor - trace* on page 4-2496
- *4.77 GICv3IRI - trace* on page 4-2504
- *4.78 GICv3InterruptTranslationService - trace* on page 4-2505
- *4.79 GICv3ProtocolChecker - trace* on page 4-2526
- *4.80 GICv3Redistributor - trace* on page 4-2527
- *4.81 GICv3RedistributorInternal - trace* on page 4-2541

About GIC600AE and GIC600AE_Filter

GIC600AE and GIC600AE_Filter are minimal models of an Arm GIC-600AE Generic Interrupt Controller, suitable for single-chip or multichip systems, where multichip operation has been tested, with the following exceptions:

- LPI/LPI command is implemented but untested.
- Power operation is unimplemented.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

They provide a simple configuration interface that allows designers to introduce GIC600AE-like functionality to their systems, while only implementing the architectural behavior, as defined by the GICv3 architecture.

All implementation-specific registers and functionality are implemented.

As with the other GIC components, there are two variants of the model with slightly different memory interfaces. Both GIC600AE and GIC600AE_Filter have a `pvbus_s` port for register accesses and a `pvbus_m` port for the LPI-related traffic from redistributors and the ITS. The `pvbus_m` port is also used to compose multichip operation.

In addition, the GIC600AE_Filter variant has a `pvbus_filtermiss_m` port, to which any transactions on the `pvbus_s` port and not directed to a 4K page used by the GIC are forwarded unmodified. Such transactions are terminated in the component when using the GIC600AE variant.

It is recommended to use the GIC600AE variant in most cases.

Also, the RAS register is implemented and various RAS errors are reported through status registers. Not all the RAS errors are available because Fast Models does not model errors that can cause ECC errors. Fast Models supports error record 0 and error record 13+. FMU registers are also implemented and are accessible by software.

Note

- Set the `FASTSIM_GIC_MEMORY_MAP` environment variable to 1 to print to `stderr` the memory map of any GICv3 or later models that are included in the platform being run.
- For writes to `GITS_TRANSLATE64R`, the *MSIRewriter* on page 3-104 component can be used to create the correct data transactions based on the device ID.

MasterID, ExtendedID, and UserFlags

This table shows how this model encodes information in the `MasterID`, `ExtendedID`, and `UserFlags` bus attributes:

Table 3-361 pvbus_s port

PVBus attribute	Bits used	Property encoded	Notes
`MasterID`	Bits[31:0]	Master ID that invoked the register access	-
`ExtendedID`	-	-	Not used.
`UserFlags`	-	-	Not used.

Note

The `pvbus_m` port does not use `MasterID`, `ExtendedID`, or `UserFlags`.

Table 3-362 Ports

Name	Protocol	Type	Description
`apb_bus`	`PVBus`	Slave	APB4 port to FMU, which will be connected from safety island for FuSa
`chip_id`	*Value* on page 2-85	Slave	chip_id port used for multichip operation
`cpu_active_s[256]`	*Signal* on page 2-85	Slave	CPUActive pins.
`fmu_error_int`	*Signal* on page 2-85	Master	FuSa FMU error interrupt signal
`fmu_fault_int`	*Signal* on page 2-85	Master	FuSa FMU fault interrupt signal
`po_reset`	*Signal* on page 2-85	Slave	Reset.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-362 Ports (continued)

Name	Protocol	Type	Description
ppi_in_0[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 0.
ppi_in_1[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 1.
ppi_in_10[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 10.
ppi_in_100[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 100.
ppi_in_101[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 101.
ppi_in_102[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 102.
ppi_in_103[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 103.
ppi_in_104[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 104.
ppi_in_105[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 105.
ppi_in_106[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 106.
ppi_in_107[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 107.
ppi_in_108[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 108.
ppi_in_109[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 109.
ppi_in_11[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 11.
ppi_in_110[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 110.
ppi_in_111[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 111.
ppi_in_112[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 112.
ppi_in_113[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 113.
ppi_in_114[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 114.
ppi_in_115[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 115.
ppi_in_116[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 116.
ppi_in_117[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 117.
ppi_in_118[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 118.
ppi_in_119[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 119.
ppi_in_12[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 12.
ppi_in_120[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 120.
ppi_in_121[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 121.
ppi_in_122[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 122.
ppi_in_123[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 123.
ppi_in_124[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 124.
ppi_in_125[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 125.
ppi_in_126[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 126.
ppi_in_127[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 127.
ppi_in_128[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 128.
ppi_in_129[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 129.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-362 Ports (continued)

Name	Protocol	Type	Description
ppi_in_13[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 13.
ppi_in_130[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 130.
ppi_in_131[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 131.
ppi_in_132[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 132.
ppi_in_133[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 133.
ppi_in_134[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 134.
ppi_in_135[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 135.
ppi_in_136[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 136.
ppi_in_137[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 137.
ppi_in_138[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 138.
ppi_in_139[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 139.
ppi_in_14[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 14.
ppi_in_140[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 140.
ppi_in_141[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 141.
ppi_in_142[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 142.
ppi_in_143[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 143.
ppi_in_144[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 144.
ppi_in_145[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 145.
ppi_in_146[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 146.
ppi_in_147[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 147.
ppi_in_148[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 148.
ppi_in_149[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 149.
ppi_in_15[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 15.
ppi_in_150[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 150.
ppi_in_151[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 151.
ppi_in_152[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 152.
ppi_in_153[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 153.
ppi_in_154[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 154.
ppi_in_155[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 155.
ppi_in_156[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 156.
ppi_in_157[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 157.
ppi_in_158[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 158.
ppi_in_159[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 159.
ppi_in_16[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 16.
ppi_in_160[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 160.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-362 Ports (continued)

Name	Protocol	Type	Description
ppi_in_161[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 161.
ppi_in_162[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 162.
ppi_in_163[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 163.
ppi_in_164[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 164.
ppi_in_165[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 165.
ppi_in_166[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 166.
ppi_in_167[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 167.
ppi_in_168[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 168.
ppi_in_169[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 169.
ppi_in_17[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 17.
ppi_in_170[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 170.
ppi_in_171[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 171.
ppi_in_172[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 172.
ppi_in_173[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 173.
ppi_in_174[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 174.
ppi_in_175[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 175.
ppi_in_176[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 176.
ppi_in_177[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 177.
ppi_in_178[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 178.
ppi_in_179[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 179.
ppi_in_18[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 18.
ppi_in_180[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 180.
ppi_in_181[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 181.
ppi_in_182[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 182.
ppi_in_183[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 183.
ppi_in_184[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 184.
ppi_in_185[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 185.
ppi_in_186[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 186.
ppi_in_187[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 187.
ppi_in_188[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 188.
ppi_in_189[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 189.
ppi_in_19[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 19.
ppi_in_190[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 190.
ppi_in_191[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 191.
ppi_in_192[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 192.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-362 Ports (continued)

Name	Protocol	Type	Description
ppi_in_193[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 193.
ppi_in_194[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 194.
ppi_in_195[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 195.
ppi_in_196[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 196.
ppi_in_197[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 197.
ppi_in_198[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 198.
ppi_in_199[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 199.
ppi_in_2[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 2.
ppi_in_20[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 20.
ppi_in_200[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 200.
ppi_in_201[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 201.
ppi_in_202[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 202.
ppi_in_203[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 203.
ppi_in_204[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 204.
ppi_in_205[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 205.
ppi_in_206[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 206.
ppi_in_207[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 207.
ppi_in_208[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 208.
ppi_in_209[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 209.
ppi_in_21[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 21.
ppi_in_210[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 210.
ppi_in_211[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 211.
ppi_in_212[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 212.
ppi_in_213[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 213.
ppi_in_214[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 214.
ppi_in_215[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 215.
ppi_in_216[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 216.
ppi_in_217[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 217.
ppi_in_218[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 218.
ppi_in_219[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 219.
ppi_in_22[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 22.
ppi_in_220[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 220.
ppi_in_221[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 221.
ppi_in_222[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 222.
ppi_in_223[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 223.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-362 Ports (continued)

Name	Protocol	Type	Description
ppi_in_224[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 224.
ppi_in_225[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 225.
ppi_in_226[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 226.
ppi_in_227[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 227.
ppi_in_228[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 228.
ppi_in_229[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 229.
ppi_in_23[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 23.
ppi_in_230[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 230.
ppi_in_231[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 231.
ppi_in_232[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 232.
ppi_in_233[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 233.
ppi_in_234[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 234.
ppi_in_235[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 235.
ppi_in_236[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 236.
ppi_in_237[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 237.
ppi_in_238[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 238.
ppi_in_239[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 239.
ppi_in_24[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 24.
ppi_in_240[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 240.
ppi_in_241[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 241.
ppi_in_242[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 242.
ppi_in_243[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 243.
ppi_in_244[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 244.
ppi_in_245[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 245.
ppi_in_246[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 246.
ppi_in_247[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 247.
ppi_in_248[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 248.
ppi_in_249[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 249.
ppi_in_25[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 25.
ppi_in_250[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 250.
ppi_in_251[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 251.
ppi_in_252[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 252.
ppi_in_253[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 253.
ppi_in_254[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 254.
ppi_in_255[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 255.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-362 Ports (continued)

Name	Protocol	Type	Description
ppi_in_26[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 26.
ppi_in_27[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 27.
ppi_in_28[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 28.
ppi_in_29[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 29.
ppi_in_3[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 3.
ppi_in_30[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 30.
ppi_in_31[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 31.
ppi_in_32[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 32.
ppi_in_33[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 33.
ppi_in_34[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 34.
ppi_in_35[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 35.
ppi_in_36[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 36.
ppi_in_37[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 37.
ppi_in_38[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 38.
ppi_in_39[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 39.
ppi_in_4[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 4.
ppi_in_40[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 40.
ppi_in_41[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 41.
ppi_in_42[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 42.
ppi_in_43[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 43.
ppi_in_44[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 44.
ppi_in_45[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 45.
ppi_in_46[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 46.
ppi_in_47[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 47.
ppi_in_48[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 48.
ppi_in_49[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 49.
ppi_in_5[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 5.
ppi_in_50[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 50.
ppi_in_51[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 51.
ppi_in_52[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 52.
ppi_in_53[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 53.
ppi_in_54[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 54.
ppi_in_55[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 55.
ppi_in_56[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 56.
ppi_in_57[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 57.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-362 Ports (continued)

Name	Protocol	Type	Description
ppi_in_58[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 58.
ppi_in_59[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 59.
ppi_in_6[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 6.
ppi_in_60[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 60.
ppi_in_61[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 61.
ppi_in_62[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 62.
ppi_in_63[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 63.
ppi_in_64[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 64.
ppi_in_65[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 65.
ppi_in_66[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 66.
ppi_in_67[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 67.
ppi_in_68[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 68.
ppi_in_69[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 69.
ppi_in_7[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 7.
ppi_in_70[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 70.
ppi_in_71[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 71.
ppi_in_72[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 72.
ppi_in_73[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 73.
ppi_in_74[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 74.
ppi_in_75[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 75.
ppi_in_76[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 76.
ppi_in_77[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 77.
ppi_in_78[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 78.
ppi_in_79[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 79.
ppi_in_8[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 8.
ppi_in_80[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 80.
ppi_in_81[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 81.
ppi_in_82[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 82.
ppi_in_83[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 83.
ppi_in_84[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 84.
ppi_in_85[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 85.
ppi_in_86[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 86.
ppi_in_87[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 87.
ppi_in_88[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 88.
ppi_in_89[16]	Signal on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 89.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-362 Ports (continued)

Name	Protocol	Type	Description
ppi_in_9[16]	*Signal* on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 9.
ppi_in_90[16]	*Signal* on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 90.
ppi_in_91[16]	*Signal* on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 91.
ppi_in_92[16]	*Signal* on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 92.
ppi_in_93[16]	*Signal* on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 93.
ppi_in_94[16]	*Signal* on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 94.
ppi_in_95[16]	*Signal* on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 95.
ppi_in_96[16]	*Signal* on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 96.
ppi_in_97[16]	*Signal* on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 97.
ppi_in_98[16]	*Signal* on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 98.
ppi_in_99[16]	*Signal* on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 99.
pvbus_filtermiss_m	PVBus	Master	Memory bus out. Transactions not filtered by the component.
pvbus_m	PVBus	Master	Memory bus out: transactions generated by the IRI.
pvbus_s	PVBus	Slave	Memory bus in: memory-mapped register accesses are accepted on this interface.
redistributor_m[256]	*GICv3Comms* on page 2-84	Master	Input from and output to CPU interface.
reset	*Signal* on page 2-85	Slave	Reset.
spi_in[960]	*Signal* on page 2-85	Slave	Shared peripheral interrupts.
wake_request[256]	*Signal* on page 2-85	Master	Power management outputs.

Table 3-363 Parameters for GIC600AE

Name	Type	Default value	Description
ARE-fixed-to-one	bool	0x1	GICv2 compatibility is not supported and GICD_CTLR.ARE_* is always one
CPU-affinities	string	""	A comma separated list of dotted quads containing the affinities of all PEs connected to this IRI. If CPU-affinities-file is specified, this parameter is ignored.
CPU-affinities-file	string	""	A file containing comma separated list of dotted quads containing the affinities of all PEs connected to this IRI. If this parameter is specified, CPU-affinities parameter will be ignored even when it is given.
DS-behaviour	int	0x2	GICD_CTLR.DS field behaviour: 0:RAZ/WI, 1:RAO/WI, 2:RW
IIDR	int	0x0	GICD_IIDR and GICR_IIDR value. When the parameter is not given or is set to zero, the GIC600 model is configured to have the latest revision.
ITS-ID-bits	int	0x10	Number of interrupt bits supported by ITS.
ITS-collection-ID-bits	int	0x8	Number of collection bits supported by ITS (optional parameter, 0 => 16bits support and GITS_TYPER.CIL=0

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-363 Parameters for GIC600AE (continued)

Name	Type	Default value	Description
ITS-count	int	0x1	Number of Interrupt Translation Services to be instantiated (0=none)
ITS-device-bits	int	0x10	Number of bits supported for ITS device IDs.
PPI-count	int	0x10	Selects the number of PPI available for each PE as 8(id22-27,29,30), 12(id 20-31) or 16(id 16-31)
RAS-CFI-support	bool	0x0	If true, fault handling interrupt for corrected errors is supported. Not supported otherwise.
RAS-FI-support	bool	0x0	If true, fault handling interrupt is supported. Not supported otherwise.
RAS-UE-support	bool	0x0	If true, In-band uncorrected error reporing is supported. Not supported otherwise.
RAS-UI-support	bool	0x0	If true, error recovery interrupt for uncorrected errors is supported. Not supported otherwise.
SPI-blocks	int	0x1e	Number of SPI blocks supported by the IRI, each block contains 32 SPIs
affinity-width	string	"0.8.8.8"	A dotted quad indicating the bitwidth of fields at each affinity level
chip-id	int	0x0	Chip ID when multichip operation is enabled.
chip-select-affinity-level	int	0x3	Affinity level 2 or 3 can be used for chip select, affinity width is forced to 0 for the level and above as this model supports one chip only
direct-lpi-support	bool	0x0	Enable support for LPI operations through GICR registers
enabled	bool	0x1	Enable GICv3 functionality; when false the component is inactive.
gict-allow-ns-reset	bool	0x1	If true, non-secure read/write access to GICT register is allowed at reset. Not allowed otherwise. This emulates gict_allow_ns tie-off signal.
max-pe-on-chip	int	0x4	Maximum number of cores on any single chip. This will be used to identify the target chip and core when multichip operation is enabled.
multichip-threaded-dgi	bool	0x1	Enable sending multichip DGI messages in a separate thread. This parameter is valid only if enable-multichip-operation is set.
print-memory-map	bool	0x0	Print memory map to stdout
redistributor-group	string	""	Redistributor grouping information with affinity as JSON : {"0": ["0.0.0.0","0.0.0.1"],"1":["0.0.1.0","0.0.1.1"]} where RD with quad 0.0.0.0 and 0.0.0.1 belongs to RD group 0. All the RDs belong to one group when this parameter is not given.
redistributor-group-file	string	""	File path to redistributor grouping information with affinity as JSON. The file uses the same format as "redistributor-group" parameter.
redistributor-power-managed-by-pwrr	bool	0x1	GIC600 dedistributor power management is done by updating GICR_PWRR register
reg-base	int	0x2c010000	GIC-600 base address

3.9.25 GIC600_Filter

GIC-600 IRI implementation: Single chip validation/filter component variant limited to 265 PE. This model is written in C++.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

GIC600_Filter contains the following CADI targets:

- GIC600

GIC600_Filter contains the following MTI components:

- *4.76 GICv3Distributor - trace* on page 4-2496
- *4.77 GICv3IRI - trace* on page 4-2504
- *4.78 GICv3InterruptTranslationService - trace* on page 4-2505
- *4.79 GICv3ProtocolChecker - trace* on page 4-2526
- *4.80 GICv3Redistributor - trace* on page 4-2527
- *4.81 GICv3RedistributorInternal - trace* on page 4-2541

About GIC600 and GIC600_Filter

GIC600 and GIC600_Filter are minimal models of an Arm GIC-600 Generic Interrupt Controller, suitable for single-chip or multichip systems, where multichip operation has been tested, with the following exceptions:

- LPI/LPI command is implemented but untested.
- Power operation is unimplemented.

They provide a simple configuration interface that allows designers to introduce GIC600-like functionality to their systems, while only implementing the architectural behavior, as defined by the GICv3 architecture.

All implementation-specific registers and functionality are implemented.

As with the other GIC components, there are two variants of the model with slightly different memory interfaces. Both GIC600 and GIC600_Filter have a `pvbus_s` port for register accesses and a `pvbus_m` port for the LPI-related traffic from redistributors and the ITS. The `pvbus_m` port is also used to compose multichip operation.

In addition, the GIC600_Filter variant has a `pvbus_filtermiss_m` port, to which any transactions on the `pvbus_s` port and not directed to a 4K page used by the GIC are forwarded unmodified. Such transactions are terminated in the component when using the GIC600 variant.

It is recommended to use the GIC600 variant in most cases.

Also, the RAS register is implemented and various RAS errors are reported through status registers. Not all the RAS errors are available because Fast Models does not model errors that can cause ECC errors. Fast Models supports error record 0 and error record 13+.

Note

- Set the `FASTSIM_GIC_MEMORY_MAP` environment variable to 1 to print to `stderr` the memory map of any GICv3 or later models that are included in the platform being run.
- For writes to `GITS_TRANSLATE64R`, the *MSIRewriter* on page 3-104 component can be used to create the correct data transactions based on the device ID.

MasterID, ExtendedID, and UserFlags

This table shows how this model encodes information in the `MasterID`, `ExtendedID`, and `UserFlags` bus attributes:

Table 3-364 pvbus_s port

PVBus attribute	Bits used	Property encoded	Notes
`MasterID`	Bits[31:0]	Master ID that invoked the register access	-
`ExtendedID`	-	-	Not used.
`UserFlags`	-	-	Not used.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Note

The pvbus_m port does not use MasterID, ExtendedID, or UserFlags.

Table 3-365 Ports

Name	Protocol	Type	Description
chip_id	*Value* on page 2-85	Slave	chip_id port which is valid from GIC600 r1p2. Writing to this port for prior GIC600 version will be ignored.
cpu_active_s[256]	*Signal* on page 2-85	Slave	CPUActive pins.
po_reset	*Signal* on page 2-85	Slave	Reset.
ppi_in_*n*[16]	*Signal* on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu *n*, where *n* is in the range 0-255.
pvbus_filtermiss_m	PVBus	Master	Memory bus out. Transactions not filtered by the component.
pvbus_m	PVBus	Master	Memory bus out: transactions generated by the IRI.
pvbus_s	PVBus	Slave	Memory bus in: memory-mapped register accesses are accepted on this interface.
redistributor_m[256]	*GICv3Comms* on page 2-84	Master	Input from and output to CPU interface.
reset	*Signal* on page 2-85	Slave	Reset.
spi_in[960]	*Signal* on page 2-85	Slave	Shared peripheral interrupts.
wake_request[256]	*Signal* on page 2-85	Master	Power management outputs.

Table 3-366 Parameters for GIC600

Name	Type	Default value	Description
ARE-fixed-to-one	bool	0x1	GICv2 compatibility is not supported and GICD_CTLR.ARE_* is always one
CPU-affinities	string	""	A comma separated list of dotted quads containing the affinities of all PEs connected to this IRI. If CPU-affinities-file is specified, this parameter is ignored.
CPU-affinities-file	string	""	A file containing comma separated list of dotted quads containing the affinities of all PEs connected to this IRI. If this parameter is specified, CPU-affinities parameter will be ignored even when it is given.
DS-behaviour	int	0x2	GICD_CTLR.DS field behaviour: 0:RAZ/WI, 1:RAO/WI, 2:RW
IIDR	int	0x0	GICD_IIDR and GICR_IIDR value. When the parameter is not given or is set to zero, the GIC600 model is configured to have the latest revision.
ITS-ID-bits	int	0x10	Number of interrupt bits supported by ITS.
ITS-collection-ID-bits	int	0x8	Number of collection bits supported by ITS (optional parameter, 0 => 16bits support and GITS_TYPER.CIL=0
ITS-count	int	0x1	Number of Interrupt Translation Services to be instantiated (0=none)
ITS-device-bits	int	0x10	Number of bits supported for ITS device IDs.
PPI-count	int	0x10	Selects the number of PPI available for each PE as 8(id22-27,29,30), 12(id 20-31) or 16(id 16-31)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-366 Parameters for GIC600 (continued)

Name	Type	Default value	Description
RAS-CFI-support	bool	0x0	If true, fault handling interrupt for corrected errors is supported. Not supported otherwise.
RAS-FI-support	bool	0x0	If true, fault handling interrupt is supported. Not supported otherwise.
RAS-UE-support	bool	0x0	If true, In-band uncorrected error reporing is supported. Not supported otherwise.
RAS-UI-support	bool	0x0	If true, error recovery interrupt for uncorrected errors is supported. Not supported otherwise.
SPI-blocks	int	0x1e	Number of SPI blocks supported by the IRI, each block contains 32 SPIs
affinity-width	string	"0.8.8.8"	A dotted quad indicating the bitwidth of fields at each affinity level
chip-id	int	0x0	Chip ID when multichip operation is enabled.
chip-select-affinity-level	int	0x3	Affinity level 2 or 3 can be used for chip select, affinity width is forced to 0 for the level and above as this model supports one chip only
direct-lpi-support	bool	0x0	Enable support for LPI operations through GICR registers
enabled	bool	0x1	Enable GICv3 functionality; when false the component is inactive.
gict-allow-ns-reset	bool	0x1	If true, non-secure read/write access to GICT register is allowed at reset. Not allowed otherwise. This emulates gict_allow_ns tie-off signal.
max-pe-on-chip	int	0x4	Maximum number of cores on any single chip. This will be used to identify the target chip and core when multichip operation is enabled.
multichip-threaded-dgi	bool	0x1	Enable sending multichip DGI messages in a separate thread. This parameter is valid only if enable-multichip-operation is set.
print-memory-map	bool	0x0	Print memory map to stdout
redistributor-group	string	""	Redistributor grouping information with affinity as JSON : {"0": ["0.0.0.0","0.0.0.1"],"1":["0.0.1.0","0.0.1.1"]} where RD with quad 0.0.0.0 and 0.0.0.1 belongs to RD group 0. All the RDs belong to one group when this parameter is not given.
redistributor-group-file	string	""	File path to redistributor grouping information with affinity as JSON. The file uses the same format as "redistributor-group" parameter.
redistributor-power-managed-by-pwrr	bool	0x1	GIC600 dedistributor power management is done by updating GICR_PWRR register
reg-base	int	0x2c010000	GIC-600 base address

3.9.26 GIC_400

This model is written in C++.

GIC_400 contains the following CADI targets:

- GIC_400

GIC_400 contains the following MTI components:

- *4.70 GIC_400 - trace* on page 4-2466
- *4.107 PVBusSlave - trace* on page 4-2760

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

About GIC_400

This component is a wrapper that permits easier configuration of the v7_VGIC component that supports parameterized configuration.

The GIC-400 has several memory-mapped interfaces at the same address. The processor that is communicating with the GIC-400 banks them. The GIC-400 must be able to distinguish from which processor a transaction originates. In the hardware, the AUSER fields on AXI supply this information to the GIC-400. In the model, there is no exact equivalent to this field. However, each transaction has a `master_id` that the model can use to identify the originating processor.

Arm clusters assign the `master_id` as follows:

- Bits[31:16]: SBZ, which the GIC-400 ignores.
- Bits[5:2]: CLUSTERID.
- Bits[1:0]: cpu_id within cluster.

CLUSTERID is the 4-bit field that either a parameter on the processor sets or a value on the `clusterid` port drives. CPUID is the core number within the cluster. CLUSTERID appears in the CP15 register space as part of the MPIDR register.

The Arm architecture suggests that each cluster in the system is given a different CLUSTERID. This distinction is essential for the VGIC to identify the cluster. The parameters in the GIC-400 component permit it to construct the map of `master_id` to interface number.

Processor interfaces that the GIC-400 supports have these parameters:

- `interfaceN.cluster_id`.
- `interfaceN.core_id`.
- `interfaceN.inout_port_number_to_use`.

N is the interface number (0-7). The `cluster_id` and `core_id` tell the GIC-400 to map that cluster or core combination to interface N.

In using `inout_port_number_to_use`, the GIC-400 has some input and output ports that pair with a particular processor interface. For example:

- The `irqcpu[]` pin wires to the `irq` port of the corresponding processor.
- The `cntpnsirq` pin from the processor wires to a `cntpnsirq[]` pin on GIC-400 to transport a *Private Peripheral Interrupt* (PPI) from the processor to the GIC-400.

The `interfaceN.inout_port_number_to_use` parameter supports clusters that can have variable numbers of cores. It tells the GIC-400 that to send to or receive a signal from the processor that is attached to interface N, it must use these pins:

- `irqout[interfaceN.inout_port_number_to_use]`.
- `fiqout[interfaceN.inout_port_number_to_use]`.
- `virqout[interfaceN.inout_port_number_to_use]`.
- `vfiqout[interfaceN.inout_port_number_to_use]`.
- `legacyirq[interfaceN.inout_port_number_to_use]`.
- `cntpnsirq[interfaceN.inout_port_number_to_use]`.
- `cntpsirq[interfaceN.inout_port_number_to_use]`.
- `legacyfiq[interfaceN.inout_port_number_to_use]`.
- `cntvirq[interfaceN.inout_port_number_to_use]`.
- `cnthpirq[interfaceN.inout_port_number_to_use]`.
- ...

`legacyirq` and `legacyfiq` are not signals from the processor but are signals into the GIC-400 from the legacy interrupt system. They are wired to PPIs. If the control registers of the GIC-400 are set up in particular ways, they can also bypass the GIC-400. See *ARM Generic Interrupt Controller Architecture version 2.0 Architecture Specification* for more information.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

The fabric between the clusters and the GIC might remap the `master_id` of a transaction. If so, then the GIC might lose the ability to identify the originating processor. The fabrics that ARM ships in Fast Models perform no such transformation.

The comparison that the GIC-400 performs on the `master_id` is only on the bottom 6 bits of the `master_id`. It ignores the rest. If you are writing your own fabric and do not properly propagate the `master_id` or transform it, the GIC-400 might not be able to identify the processor. The source code for the GIC_400 component can be examined to see how it might be adapted for it to understand different `master_id` schemes.

Differences between the model and the RTL

The GIC-400 model has these limitations:

- Reads and writes to GICD_ISACTIVERn/GICD_ICACTIVERn/GICD_ISPENDRn/ GICD_ICPENDRn might not work as expected unless there is a configured target in GICD_ICFGRm.
- Some of the interaction of GICD_CTLR.EnableGrpX and level sensitive interrupts might not work correctly.
- It does not model the signals nIRQOUT/nFIQOUT.
- It models interrupts with positive logic, rather than the negative logic that the hardware uses. Hence, the signal pins omit the “n” prefix in their names.

Table 3-367 Ports

Name	Protocol	Type	Description
`cfgsdisable`	*Signal* on page 2-85	Slave	Disable write access to some GIC registers.
`cnthpirq[8]`	*Signal* on page 2-85	Slave	Secure physical timer event. PPI interrupt id 26.
`cntpnsirq[8]`	*Signal* on page 2-85	Slave	Non-secure physical timer event. PPI interrupt id 30.
`cntpsirq[8]`	*Signal* on page 2-85	Slave	Secure physical timer event. PPI interrupt id 29.
`cntvirq[8]`	*Signal* on page 2-85	Slave	Virtual timer event. PPI interrupt id 27.
`fiqcpu[8]`	*Signal* on page 2-85	Master	FIQ signal to the corresponding processor.
`fiqout[8]`	*Signal* on page 2-85	Master	FIQOUT signal to the corresponding processor.
`irqcpu[8]`	*Signal* on page 2-85	Master	IRQ signal to the corresponding processor.
`irqout[8]`	*Signal* on page 2-85	Master	IRQOUT signal to the corresponding processor.
`irqs[480]`	*Signal* on page 2-85	Slave	Interrupt request input lines for the GIC.
`legacyfiq[8]`	*Signal* on page 2-85	Slave	Signal into the GIC-400 from the legacy interrupt system. PPI interrupt id 28.
`legacyirq[8]`	*Signal* on page 2-85	Slave	Signal into the GIC-400 from the legacy interrupt system. PPI interrupt id 31.
`pvbus_s`	`PVBus`	Slave	Handles incoming transactions from PVBus masters.
`reset_signal`	*Signal* on page 2-85	Slave	Reset signal input.
`vfiqcpu[8]`	*Signal* on page 2-85	Master	Virtual FIQ signal to the processor.
`virqcpu[8]`	*Signal* on page 2-85	Master	Virtual IRQ signal to the processor.

Table 3-368 Parameters for GIC_400

Name	Type	Default value	Description
`enable_log_errors`	`bool`	`0x0`	Enable logging of errors
`enable_log_fatal`	`bool`	`0x0`	Enable logging of fatal errors

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-368 Parameters for GIC_400 (continued)

Name	Type	Default value	Description
enable_log_warnings	bool	0x0	Enable logging of warnings
enabled	bool	0x1	Enable the component. If it is disabled then all register writes will have no effect.

3.9.27 GIC_IRI

GIC metacomponent for redistribution of interrupts(contains Distributor, and configurable numbers of ITSs and Redistributors. This model is written in C++.

GIC_IRI contains the following CADI targets:

- GIC_IRI

GIC_IRI contains the following MTI components:

- *4.76 GICv3Distributor - trace* on page 4-2496
- *4.77 GICv3IRI - trace* on page 4-2504
- *4.80 GICv3Redistributor - trace* on page 4-2527
- *4.81 GICv3RedistributorInternal - trace* on page 4-2541

Table 3-369 Ports

Name	Protocol	Type	Description
cfgsdisable	*Signal* on page 2-85	Slave	Disable some SPIs signal.
extended_ppi_in_*n*[64]	*Signal* on page 2-85	Slave	Extended private peripheral interrupts (ID1056-ID1119) for cpu *n*, where *n* is in the range 0-255.
extended_spi_in[1024]	*Signal* on page 2-85	Slave	Extended Shared peripheral interrupts.
msi_error_interrupt	*Signal* on page 2-85	Master	When report of MSI error allowed through interrupt, loop-back this signal to relevant IRQ input signal
po_reset	*Signal* on page 2-85	Slave	Resets.
ppi_in_*n*[16]	*Signal* on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu *n*, where *n* is in the range 0-255.
pvbus_m	PVBus	Master	Memory bus out: transactions generated by the IRI.
pvbus_s	PVBus	Slave	Memory bus in: memory-mapped register accesses are accepted on this interface.
redistributor_m[256]	*GICv3Comms* on page 2-84	Master	Input from and output to CPU interface.
reset	*Signal* on page 2-85	Slave	Resets.
spi_in[988]	*Signal* on page 2-85	Slave	Shared peripheral interrupts.
wake_request[256]	*Signal* on page 2-85	Master	Power management outputs.
wire_to_msi_0[1024]	*Signal* on page 2-85	Slave	Wire-to-MSI interrupts for architectural consolidator 0.
wire_to_msi_1[1024]	*Signal* on page 2-85	Slave	Wire-to-MSI interrupts for architectural consolidator 1.
wire_to_msi_2[1024]	*Signal* on page 2-85	Slave	Wire-to-MSI interrupts for architectural consolidator 2.
wire_to_msi_3[1024]	*Signal* on page 2-85	Slave	Wire-to-MSI interrupts for architectural consolidator 3.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-370 Parameters for GIC_IRI

Name	Type	Default value	Description
A3-affinity-supported	bool	0x0	Device supports affinity level 3 values that are non-zero.
ARE-fixed-to-one	bool	0x0	GICv2 compatibility is not supported and GICD_CTLR.ARE_* is always one
CPU-affinities	string	""	A comma separated list of dotted quads containing the affinities of all PEs connected to this IRI. If CPU-affinities-file is specified, this parameter is ignored.
CPU-affinities-file	string	""	A file containing comma separated list of dotted quads containing the affinities of all PEs connected to this IRI. If this parameter is specified, CPU-affinities parameter will be ignored even when it is given.
DPG-ARE-only	bool	0x0	Limit application of DPG bits to interrupt groups for which ARE=1
DPG-bits-implemented	bool	0x0	Enable implementation of interrupt group participation bits or DPG bits in GICR_CTLR
DS-fixed-to-zero	bool	0x0	Enable/disable support of single security state
GICD-alias	int	0x0	In GICv2 mode: the base address for a 4k page alias of the first 4k of the Distributor page, in GICv3 mode. the base address of a 64KB page containing message based SPI signalling register aliases(0:Disabled)
GICD-legacy-registers-as-reserved	bool	0x0	When ARE is RAO/WI, makes superfluous registers in GICD reserved (including for the purpose of STATUSR updates)
GICD_CTLR-DS-1-means-secure-only	bool	0x0	If GICD_CTLR.DS=1, GICD supports a single security state which is secure if this is true, otherwise is non-secure.
GICD_ITARGETSR-RAZWI	bool	0x0	If true, the GICD_ITARGETS registers are RAZ/WI
GICD_PIDR	int	0x0	The value for the GICD_PIDR registers, if non-zero. Note: fixed fields (device type etc.) will be overriden in this value.
GICR-clear-enable-supported	bool	0x0	When true, this sets the value of the RO bit GICR_CTLR.CES with the value of the parameter allow-LPIEN-clear, making it visible to software
GICR-invalidate-registers-implemented	bool	0x0	When true, the registers GICR_INVLPIR, GICR_INVALLR and GICR_SYNCR are implemented
GICR_PIDR	int	0x0	The value for the GICR_PIDR registers, if non-zero. Note: fixed fields (device type etc.) will be overriden in this value.
GICR_PROPBASER-read-only	bool	0x0	GICR_PROPBASER register is read-only.
GICR_PROPBASER-reset-value	int	0x0	Value of GICR_PROPBASER on reset.
GITS_BASER0-entry-bytes	int	0x8	Number of bytes required per entry for GITS_BASER0 register.
GITS_BASER0-indirect-RAZ	bool	0x0	Indirect field for GITS_BASER0 register is RAZ/WI.
GITS_BASER0-type	int	0x0	Type field for GITS_BASER0 register. 0 = Unimplemented; 1 = Devices; 2 = Virtual Processors; 3 = Physical Processors; 4 = Collections
GITS_BASER1-entry-bytes	int	0x8	Number of bytes required per entry for GITS_BASER1 register.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-370 Parameters for GIC_IRI (continued)

Name	Type	Default value	Description
GITS_BASER1-indirect-RAZ	bool	0x0	Indirect field for GITS_BASER1 register is RAZ/WI.
GITS_BASER1-type	int	0x0	Type field for GITS_BASER1 register. 0 = Unimplemented; 1 = Devices; 2 = Virtual Processors; 3 = Physical Processors; 4 = Collections
GITS_BASER2-entry-bytes	int	0x8	Number of bytes required per entry for GITS_BASER2 register.
GITS_BASER2-indirect-RAZ	bool	0x0	Indirect field for GITS_BASER2 register is RAZ/WI.
GITS_BASER2-type	int	0x0	Type field for GITS_BASER2 register. 0 = Unimplemented; 1 = Devices; 2 = Virtual Processors; 3 = Physical Processors; 4 = Collections
GITS_BASER3-entry-bytes	int	0x8	Number of bytes required per entry for GITS_BASER3 register.
GITS_BASER3-indirect-RAZ	bool	0x0	Indirect field for GITS_BASER3 register is RAZ/WI.
GITS_BASER3-type	int	0x0	Type field for GITS_BASER3 register. 0 = Unimplemented; 1 = Devices; 2 = Virtual Processors; 3 = Physical Processors; 4 = Collections
GITS_BASER4-entry-bytes	int	0x8	Number of bytes required per entry for GITS_BASER4 register.
GITS_BASER4-indirect-RAZ	bool	0x0	Indirect field for GITS_BASER4 register is RAZ/WI.
GITS_BASER4-type	int	0x0	Type field for GITS_BASER4 register. 0 = Unimplemented; 1 = Devices; 2 = Virtual Processors; 3 = Physical Processors; 4 = Collections
GITS_BASER5-entry-bytes	int	0x8	Number of bytes required per entry for GITS_BASER5 register.
GITS_BASER5-indirect-RAZ	bool	0x0	Indirect field for GITS_BASER5 register is RAZ/WI.
GITS_BASER5-type	int	0x0	Type field for GITS_BASER5 register. 0 = Unimplemented; 1 = Devices; 2 = Virtual Processors; 3 = Physical Processors; 4 = Collections
GITS_BASER6-entry-bytes	int	0x8	Number of bytes required per entry for GITS_BASER6 register.
GITS_BASER6-indirect-RAZ	bool	0x0	Indirect field for GITS_BASER6 register is RAZ/WI.
GITS_BASER6-type	int	0x0	Type field for GITS_BASER6 register. 0 = Unimplemented; 1 = Devices; 2 = Virtual Processors; 3 = Physical Processors; 4 = Collections
GITS_BASER7-entry-bytes	int	0x8	Number of bytes required per entry for GITS_BASER7 register.
GITS_BASER7-indirect-RAZ	bool	0x0	Indirect field for GITS_BASER7 register is RAZ/WI.
GITS_BASER7-type	int	0x0	Type field for GITS_BASER7 register. 0 = Unimplemented; 1 = Devices; 2 = Virtual Processors; 3 = Physical Processors; 4 = Collections
GITS_PIDR	int	0x0	The value for the GITS_PIDR registers, if non-zero. Note: fixed fields (device type etc.) will be overriden in this value.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-370 Parameters for GIC_IRI (continued)

Name	Type	Default value	Description
ICFGR-PPI-mask	int	0xaaaaaaaa	Mask for writes to ICFGR registers that configure PPIs
ICFGR-PPI-reset	int	0x0	Reset value for ICFGR regesters that configure PPIs
ICFGR-SGI-mask	int	0x0	Mask for writes to ICFGR registers that configure SGIs
ICFGR-SGI-reset	int	0xaaaaaaaa	Reset value for ICFGR registers that configure SGIs
ICFGR-SPI-mask	int	0xaaaaaaaa	Mask for writes to ICFGR registers that configure SPIs
ICFGR-SPI-reset	int	0x0	Reset value for ICFGR regesters that configure SPIs
ICFGR-rsvd-bit	bool	0x0	If ARE=0, the value of reserved bits i.e. bit 0,2,4..30 of ICFGRn for n>0
IGROUP-PPI-mask	int	0xffff	Mask for writes to PPI bits in IGROUP registers
IGROUP-PPI-reset	int	0x0	Reset value for SGI bits in IGROUP registers
IGROUP-SGI-mask	int	0xffff	Mask for writes to SGI bits in IGROUP registers
IGROUP-SGI-reset	int	0x0	Reset value for SGI bits in IGROUP registers
IIDR	int	0x0	GICD_IIDR and GICR_IIDR value
IRI-ID-bits	int	0x10	Number of bits used to represent interrupts IDs in the Distributor and Redistributors, forced to 10 when none of LPIs, extended SPIs or extended PPIs is supported
IROUTER-IRM-RAZ-WI	bool	0x0	GICD_IROUTERn.InterruptRoutingMode is RAZ/WI
ITS-BASER-force-page-alignement	bool	0x1	Force alignement of address writen to a GITS_BASER register to the page size configured
ITS-ID-bits	int	0x10	Number of interrupt bits supported by ITS.
ITS-MOVALL-update-collections	bool	0x0	Whether MOVALL command updates the collection entires
ITS-TRANSLATE64R	bool	0x0	Add an implementation specific register at 0x10008 supporting 64 bit TRANSLATER (dev[63:32], interupt[31:0])
ITS-collection-ID-bits	int	0x0	Number of collection bits supported by ITS (optional parameter, 0 => 16bits support and GITS_TYPER.CIL=0
ITS-count	int	0x0	Number of Interrupt Translation Services to be instantiated (0=none)
ITS-cumulative-collection-tables	bool	0x1	When true, the supported amount of collections is the sum of GITS_TYPER.HCC and the number of collections supported in memory, otherwise, simply the number supported in memory only. Irrelevant when HCC=0
ITS-device-bits	int	0x10	Number of bits supported for ITS device IDs.
ITS-entry-size	int	0x8	Number of bytes required to store each entry in the ITT tables.
ITS-hardware-collection-count	int	0x0	Number of hardware collections held exclusively in the ITS
ITS-legacy-iidr-typer-offset	bool	0x0	Put the GITS_IIDR and GITS_TYPER registers at their older offset of 0x8 and 0x4 respectively
ITS-shared-vPE-table	int	0x0	Number of affinity levels to which the vPE configuration table is shared. This parameter is valid when has-gicv4.1 is true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-370 Parameters for GIC_IRI (continued)

Name	Type	Default value	Description
ITS-threaded-command-queue	bool	0x1	Enable execution of ITS commands in a separate thread which is sometimes required for cosimulation
ITS-use-physical-target-addresses	bool	0x1	Use physical hardware adresses for targets in ITS commands -- must be true for distributed implementations
ITS-vmovp-bit	bool	0x0	Device supports software issuing a VMOVP to only one of the ITSs that has a mapping for a vPE. The device itself ensures synchronization of the VMOVP command across all ITSs that have mapping for that vPE.
ITS0-base	int	0x0	Register base address for ITS0 (automatic if 0).
ITS1-base	int	0x0	Register base address for ITS1 (automatic if 0).
ITS2-base	int	0x0	Register base address for ITS2 (automatic if 0).
ITS3-base	int	0x0	Register base address for ITS3 (automatic if 0).
LPI-cache-check-data	bool	0x0	Enable Cached LPI data against memory checking when available for cache type
LPI-cache-type	int	0x1	Cache type for LPIs, 0:No caching, 1:Full caching
MSI_IIDR	int	0x0	Value returned in MSI_IIDR registers.
MSI_NS-frame0-base	int	0x0	If non-zero, sets the base address used for non-secure MSI frame 0 registers.
MSI_NS-frame0-max-SPI	int	0x0	Maximum SPI ID supported by non-secure MSI frame 0. Set to 0 to disable frame.
MSI_NS-frame0-min-SPI	int	0x0	Minimum SPI ID supported by non-secure MSI frame 0. Set to 0 to disable frame.
MSI_NS-frame1-base	int	0x0	If non-zero, sets the base address used for non-secure MSI frame 1 registers.
MSI_NS-frame1-max-SPI	int	0x0	Maximum SPI ID supported by non-secure MSI frame 1. Set to 0 to disable frame.
MSI_NS-frame1-min-SPI	int	0x0	Minimum SPI ID supported by non-secure MSI frame 1. Set to 0 to disable frame.
MSI_NS-frame2-base	int	0x0	If non-zero, sets the base address used for non-secure MSI frame 2 registers.
MSI_NS-frame2-max-SPI	int	0x0	Maximum SPI ID supported by non-secure MSI frame 2. Set to 0 to disable frame.
MSI_NS-frame2-min-SPI	int	0x0	Minimum SPI ID supported by non-secure MSI frame 2. Set to 0 to disable frame.
MSI_NS-frame3-base	int	0x0	If non-zero, sets the base address used for non-secure MSI frame 3 registers.
MSI_NS-frame3-max-SPI	int	0x0	Maximum SPI ID supported by non-secure MSI frame 3. Set to 0 to disable frame.
MSI_NS-frame3-min-SPI	int	0x0	Minimum SPI ID supported by non-secure MSI frame 3. Set to 0 to disable frame.
MSI_NS-frame4-base	int	0x0	If non-zero, sets the base address used for non-secure MSI frame 4 registers.
MSI_NS-frame4-max-SPI	int	0x0	Maximum SPI ID supported by non-secure MSI frame 4. Set to 0 to disable frame.
MSI_NS-frame4-min-SPI	int	0x0	Minimum SPI ID supported by non-secure MSI frame 4. Set to 0 to disable frame.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-370 Parameters for GIC_IRI (continued)

Name	Type	Default value	Description
MSI_NS-frame5-base	int	0x0	If non-zero, sets the base address used for non-secure MSI frame 5 registers.
MSI_NS-frame5-max-SPI	int	0x0	Maximum SPI ID supported by non-secure MSI frame 5. Set to 0 to disable frame.
MSI_NS-frame5-min-SPI	int	0x0	Minimum SPI ID supported by non-secure MSI frame 5. Set to 0 to disable frame.
MSI_NS-frame6-base	int	0x0	If non-zero, sets the base address used for non-secure MSI frame 6 registers.
MSI_NS-frame6-max-SPI	int	0x0	Maximum SPI ID supported by non-secure MSI frame 6. Set to 0 to disable frame.
MSI_NS-frame6-min-SPI	int	0x0	Minimum SPI ID supported by non-secure MSI frame 6. Set to 0 to disable frame.
MSI_NS-frame7-base	int	0x0	If non-zero, sets the base address used for non-secure MSI frame 7 registers.
MSI_NS-frame7-max-SPI	int	0x0	Maximum SPI ID supported by non-secure MSI frame 7. Set to 0 to disable frame.
MSI_NS-frame7-min-SPI	int	0x0	Minimum SPI ID supported by non-secure MSI frame 7. Set to 0 to disable frame.
MSI_PIDR	int	0x0	The value for the MSI_PIDR registers, if non-zero and distributor supports GICv2m. Note: fixed fields (device type etc.) will be overriden in this value.
MSI_S-frame0-base	int	0x0	If non-zero, sets the base address used for secure MSI frame 0 registers.
MSI_S-frame0-max-SPI	int	0x0	Maximum SPI ID supported by secure MSI frame 0. Set to 0 to disable frame.
MSI_S-frame0-min-SPI	int	0x0	Minimum SPI ID supported by secure MSI frame 0. Set to 0 to disable frame.
MSI_S-frame1-base	int	0x0	If non-zero, sets the base address used for secure MSI frame 1 registers.
MSI_S-frame1-max-SPI	int	0x0	Maximum SPI ID supported by secure MSI frame 1. Set to 0 to disable frame.
MSI_S-frame1-min-SPI	int	0x0	Minimum SPI ID supported by secure MSI frame 1. Set to 0 to disable frame.
MSI_S-frame2-base	int	0x0	If non-zero, sets the base address used for secure MSI frame 2 registers.
MSI_S-frame2-max-SPI	int	0x0	Maximum SPI ID supported by secure MSI frame 2. Set to 0 to disable frame.
MSI_S-frame2-min-SPI	int	0x0	Minimum SPI ID supported by secure MSI frame 2. Set to 0 to disable frame.
MSI_S-frame3-base	int	0x0	If non-zero, sets the base address used for secure MSI frame 3 registers.
MSI_S-frame3-max-SPI	int	0x0	Maximum SPI ID supported by secure MSI frame 3. Set to 0 to disable frame.
MSI_S-frame3-min-SPI	int	0x0	Minimum SPI ID supported by secure MSI frame 3. Set to 0 to disable frame.
MSI_S-frame4-base	int	0x0	If non-zero, sets the base address used for secure MSI frame 4 registers.
MSI_S-frame4-max-SPI	int	0x0	Maximum SPI ID supported by secure MSI frame 4. Set to 0 to disable frame.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-370 Parameters for GIC_IRI (continued)

Name	Type	Default value	Description
MSI_S-frame4-min-SPI	int	0x0	Minimum SPI ID supported by secure MSI frame 4. Set to 0 to disable frame.
MSI_S-frame5-base	int	0x0	If non-zero, sets the base address used for secure MSI frame 5 registers.
MSI_S-frame5-max-SPI	int	0x0	Maximum SPI ID supported by secure MSI frame 5. Set to 0 to disable frame.
MSI_S-frame5-min-SPI	int	0x0	Minimum SPI ID supported by secure MSI frame 5. Set to 0 to disable frame.
MSI_S-frame6-base	int	0x0	If non-zero, sets the base address used for secure MSI frame 6 registers.
MSI_S-frame6-max-SPI	int	0x0	Maximum SPI ID supported by secure MSI frame 6. Set to 0 to disable frame.
MSI_S-frame6-min-SPI	int	0x0	Minimum SPI ID supported by secure MSI frame 6. Set to 0 to disable frame.
MSI_S-frame7-base	int	0x0	If non-zero, sets the base address used for secure MSI frame 7 registers.
MSI_S-frame7-max-SPI	int	0x0	Maximum SPI ID supported by secure MSI frame 7. Set to 0 to disable frame.
MSI_S-frame7-min-SPI	int	0x0	Minimum SPI ID supported by secure MSI frame 7. Set to 0 to disable frame.
PA_SIZE	int	0x30	Number of valid bits in physical address
PPI-implemented-mask	int	0xffff	Mask of PPIs that are implemented. One bit per PPI bit 0 == PPI 16 (first PPI). This will affect other masks.
SPI-count	int	0xe0	Number of SPIs that are implemented.
SPI-message-based-support	bool	0x1	Distributor supports meassage based signaling of SPI
SPI-unimplemented	string	""	A comma spearated list of unimplemented SPIs ranges for sparse SPI defintion(for ex: '35, 39-42, 73)'
STATUSR-implemented	bool	0x1	Determines whether the GICR_STATUSR register is implemented.
allow-LPIEN-clear	bool	0x0	Allow RW behaviour on GICR_CTLR.LPIEN isntead of set once
chip-id	int	0x0	Chip ID when multichip operation is enabled.
common-lpi-configuration	int	0x0	Describes which re-distributors share (and must be configured with the same) LPI configuration table as described in GICR_TYPER(0:All, 1:A.x.x.x, 2:A.B.x.x, 3:A.B.C.x
common-vPE-table-affinity	string	""	Affinity value list in the form of 'a.b.c.d, e.f.g.h, etc' given to the ITS(s) (where a.b.c.d corresponds to ITS0, e.f.g.h corresponds to ITS1 and so on). Under an affinity value the vPE configuration table is shared among redistributors where the level to be shared is defined by ITS-shared-vPE-table. This parameter is valid when has-gicv4.1 is true.
consolidators	string	""	Specify consolidators' base addresses, interrupt line counts and base interrupt IDs, in the form 'baseAddr0:itlineCount0:baseINTID0, baseAddr1:itlineCount1:baseINTID1, [etc]' (eg '0x3f100000:64:4096, 0x3f200000:64:4224'). The consolidators' count is inferred from the list (maximum of 4). If not specified, the component contains no consolidators.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-370 Parameters for GIC_IRI (continued)

Name	Type	Default value	Description
delay-ITS-accesses	bool	0x1	Delay accesses from the ITS until GICR_SYNCR is read.
delay-redistributor-accesses	bool	0x1	Delay memory accesses from the redistributor until GICR_SYNCR is read.
direct-lpi-support	bool	0x0	Enable support for LPI operations through GICR registers
enable-multichip-operation	bool	0x0	Enables multi-chip operation between Distributors in distributed GIC IRI
enable_protocol_checking	bool	0x0	Enable/disable protocol checking at cpu interface
enabled	bool	0x1	Enable GICv3 functionality; when false the component is inactive.
extended-ppi-count	int	0x0	Number of extended PPI supported
extended-spi-count	int	0x0	Number of extended SPI supported
fixed-routed-spis	string	""	Value of IROUTER[n] register in the form 'n=a.b.c.d, n=*'. The RM bit of IROUTER is 0 when n=a.b.c.d is used else 1 when n=* is used. n can be >= 32 and <= 1019
gicv2-only	bool	0x0	If true, when using the GICv3 model, pretend to be a GICv2 system
has-two-security-states	bool	0x1	If true, has two security states
has_VPENDBASER-dirty-flag-on-load	bool	0x0	GICR_VPENDBASER.Dirty reflects transient loading state when valid=1
has_mpam	bool	0x0	Enable MPAM support on ITS and RDs
ignore-generate-sgi-when-no-are	bool	0x0	Ignore GenerateSGI packets coming form the CPU interface if both ARE_S and ARE_NS are 0
irouter-default-mask	string	""	Default Mask value for IROUTER[32..1019] register in the form 'a.b.c.d'
irouter-default-reset	string	""	Default Reset Value of IROUTER[32..1019] register in the form 'a.b.c.d' or *
irouter-mask-values	string	""	Mask Value of IROUTER[n] register in the form 'n=a.b.c.d'.n can be >= 32 and <= 1019
irouter-reset-values	string	""	Reset Value of IROUTER[n] register in the form 'n=a.b.c.d or n=*'.n can be >= 32 and <= 1019
legacy-sgi-enable-rao	bool	0x0	Enables for SGI associated with an ARE=0 regime are RAO/WI
local-SEIs	bool	0x0	Generate SEI to signal internal issues
local-VSEIs	bool	0x0	Generate VSEI to signal internal issues
lockable-SPI-count	int	0x0	Number of SPIs that are locked down when CFGSDISABLE signal is asserted. Only applies for GICv2.
max-pe-on-chip	int	0x4	Maximum number of cores on any single chip. This will be used to identify the target chip and core when multichip operation is enabled.
monolithic	bool	0x0	Indicate that the implementation is not distributed
mpam_max_partid	int	0xffff	Maximum valid PARTID
mpam_max_pmg	int	0xff	Maximum valid PMG

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-370 Parameters for GIC_IRI (continued)

Name	Type	Default value	Description
multichip-threaded-dgi	bool	0x1	Enable sending multichip DGI messages in a separate thread. This parameter is valid only if enable-multichip-operation is set.
non-ARE-core-count	int	0x8	Maximum number of non-ARE cores; normally used to pass the cluster-level NUM_CORES parameter to the top-level redistributor.
outer-cacheability-support	bool	0x0	Allow configuration of outer cachability attributes in ITS and Redistributor
output_attributes	string	"ExtendedID[62:55]=MPAM_PMG, ExtendedID[54:39]=MPAM_PARTID, ExtendedID[38]=MPAM_NS"	User-defined transform to be applied to bus attributes like MasterID, ExtendedID or UserFlags. Currently, only works for MPAM Attributes encoding into bus attributes.
print-memory-map	bool	0x0	Print memory map to stdout
priority-bits	int	0x5	Number of implemented priority bits
processor-numbers	string	""	Specify processor numbers (as appears in GICR_TYPER) in the form 0.0.0.0=0,0.0.0.1=1 etc.) If not specified, will number processors starting at 0.
redistributor-threaded-sync	bool	0x1	Enable execution of redistributor delayed transactions in a separate thread which is sometimes required for cosimulation
reg-base	int	0x2c010000	Base for decoding GICv3 registers.
reg-base-per-redistributor	string	""	Base address for each redistributor in the form '0.0.0.0=0x2c010000, 0.0.0.1=0x2c020000'. All redistributors must be specified and this overrides the reg-base parameter (except that reg-base will still be used for the top-level redistributor). If reg-base-per-redistributor-file is specified, this parameter is ignored.
reg-base-per-redistributor-file	string	""	Path to file containing the base address for each redistributor in the form '0.0.0.0=0x2c010000, 0.0.0.1=0x2c020000'. All redistributors must be specified and this overrides the reg-base parameter (except that reg-base will still be used for the top-level redistributor). If this parameter is specified, reg-base-per-redistributor parameter will be ignored even when it is given.
report-MSI-error-via-statusr	int	0x0	Report MSI error via GITS_STATUSR. (0:unsupported, 1:report by GITS_STATUSR, 2:report by GITS_STATUSR and interrupt as well).
sgi-range-selector-support	bool	0x0	Device has support for the Range Selector feature for SGI
single-set-support	bool	0x0	When true, forces redistributors to recall interrupts with a clear rather than issue a second Set command
supports-shareability	bool	0x1	Device supports shareability attributes on outgoing memory bus (i.e. is modelling an ACElite port rather than an AXI4 port).
trace-speculative-lpi-property-update	bool	0x0	Trace LPI propery updates performed on speculative accesses (useful for debuging LPI)
virtual-lpi-support	bool	0x0	GICv4 Virtual LPIs and Direct injection of Virtual LPIs supported

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-370 Parameters for GIC_IRI (continued)

Name	Type	Default value	Description
virtual-priority-bits	int	0x5	Number of implemented virtual priority bits
wakeup-on-reset	bool	0x0	Go against specification and start redistributors in woken-up state at reset. This allows software that was written for previous versions of the GICv3 specification to work correctly. This should not be used for production code or when the distributor is used separately from the core fast model.

3.9.28 GIC_IRI_Filter

GIC metacomponent for redistribution of interrupts(contains Distributor, and configurable numbers of ITSs and Redistributors. Validation only version. This model is written in C++.

GIC_IRI_Filter contains the following CADI targets:

- GIC_IRI

GIC_IRI_Filter contains the following MTI components:

- *4.76 GICv3Distributor - trace* on page 4-2496
- *4.77 GICv3IRI - trace* on page 4-2504
- *4.80 GICv3Redistributor - trace* on page 4-2527
- *4.81 GICv3RedistributorInternal - trace* on page 4-2541

Table 3-371 Ports

Name	Protocol	Type	Description
cfgsdisable	*Signal* on page 2-85	Slave	Disable some SPIs signal.
extended_ppi_in_*n*[64]	*Signal* on page 2-85	Slave	Extended private peripheral interrupts (ID1056-ID1119) for cpu *n*, where *n* is in the range 0-255.
extended_spi_in[1024]	*Signal* on page 2-85	Slave	Extended Shared peripheral interrupts.
msi_error_interrupt	*Signal* on page 2-85	Master	When report of MSI error allowed through interrupt, loop-back this signal to relevant IRQ input signal
po_reset	*Signal* on page 2-85	Slave	Resets.
ppi_in_*n*[16]	*Signal* on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu *n*, where *n* is in the range 0-255.
pvbus_filtermiss_m	PVBus	Master	Passthrough for accesses to pages not used by the GIC IRI.
pvbus_m	PVBus	Master	Memory bus for transactions generated by the GIC.
pvbus_s	PVBus	Slave	Memory bus in.
redistributor_m[256]	*GICv3Comms* on page 2-84	Master	Input from and output to CPU interface.
reset	*Signal* on page 2-85	Slave	Resets.
spi_in[988]	*Signal* on page 2-85	Slave	Shared peripheral interrupts.
wake_request[256]	*Signal* on page 2-85	Master	Power management outputs.
wire_to_msi_0[1024]	*Signal* on page 2-85	Slave	Wire-to-MSI interrupts for architectural consolidator 0.
wire_to_msi_1[1024]	*Signal* on page 2-85	Slave	Wire-to-MSI interrupts for architectural consolidator 1.
wire_to_msi_2[1024]	*Signal* on page 2-85	Slave	Wire-to-MSI interrupts for architectural consolidator 2.
wire_to_msi_3[1024]	*Signal* on page 2-85	Slave	Wire-to-MSI interrupts for architectural consolidator 3.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-372 Parameters for GIC_IRI

Name	Type	Default value	Description
A3-affinity-supported	bool	0x0	Device supports affinity level 3 values that are non-zero.
ARE-fixed-to-one	bool	0x0	GICv2 compatibility is not supported and GICD_CTLR.ARE_* is always one
CPU-affinities	string	""	A comma separated list of dotted quads containing the affinities of all PEs connected to this IRI. If CPU-affinities-file is specified, this parameter is ignored.
CPU-affinities-file	string	""	A file containing comma separated list of dotted quads containing the affinities of all PEs connected to this IRI. If this parameter is specified, CPU-affinities parameter will be ignored even when it is given.
DPG-ARE-only	bool	0x0	Limit application of DPG bits to interrupt groups for which ARE=1
DPG-bits-implemented	bool	0x0	Enable implementation of interrupt group participation bits or DPG bits in GICR_CTLR
DS-fixed-to-zero	bool	0x0	Enable/disable support of single security state
GICD-alias	int	0x0	In GICv2 mode: the base address for a 4k page alias of the first 4k of the Distributor page, in GICv3 mode. the base address of a 64KB page containing message based SPI signalling register aliases(0:Disabled)
GICD-legacy-registers-as-reserved	bool	0x0	When ARE is RAO/WI, makes superfluous registers in GICD reserved (including for the purpose of STATUSR updates)
GICD_CTLR-DS-1-means-secure-only	bool	0x0	If GICD_CTLR.DS=1, GICD supports a single security state which is secure if this is true, otherwise is non-secure.
GICD_ITARGETSR-RAZWI	bool	0x0	If true, the GICD_ITARGETS registers are RAZ/WI
GICD_PIDR	int	0x0	The value for the GICD_PIDR registers, if non-zero. Note: fixed fields (device type etc.) will be overriden in this value.
GICR-clear-enable-supported	bool	0x0	When true, this sets the value of the RO bit GICR_CTLR.CES with the value of the parameter allow-LPIEN-clear, making it visible to software
GICR-invalidate-registers-implemented	bool	0x0	When true, the registers GICR_INVLPIR, GICR_INVALLR and GICR_SYNCR are implemented
GICR_PIDR	int	0x0	The value for the GICR_PIDR registers, if non-zero. Note: fixed fields (device type etc.) will be overriden in this value.
GICR_PROPBASER-read-only	bool	0x0	GICR_PROPBASER register is read-only.
GICR_PROPBASER-reset-value	int	0x0	Value of GICR_PROPBASER on reset.
GITS_BASER0-entry-bytes	int	0x8	Number of bytes required per entry for GITS_BASER0 register.
GITS_BASER0-indirect-RAZ	bool	0x0	Indirect field for GITS_BASER0 register is RAZ/WI.
GITS_BASER0-type	int	0x0	Type field for GITS_BASER0 register. 0 = Unimplemented; 1 = Devices; 2 = Virtual Processors; 3 = Physical Processors; 4 = Collections
GITS_BASER1-entry-bytes	int	0x8	Number of bytes required per entry for GITS_BASER1 register.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-372 Parameters for GIC_IRI (continued)

Name	Type	Default value	Description
GITS_BASER1-indirect-RAZ	bool	0x0	Indirect field for GITS_BASER1 register is RAZ/WI.
GITS_BASER1-type	int	0x0	Type field for GITS_BASER1 register. 0 = Unimplemented; 1 = Devices; 2 = Virtual Processors; 3 = Physical Processors; 4 = Collections
GITS_BASER2-entry-bytes	int	0x8	Number of bytes required per entry for GITS_BASER2 register.
GITS_BASER2-indirect-RAZ	bool	0x0	Indirect field for GITS_BASER2 register is RAZ/WI.
GITS_BASER2-type	int	0x0	Type field for GITS_BASER2 register. 0 = Unimplemented; 1 = Devices; 2 = Virtual Processors; 3 = Physical Processors; 4 = Collections
GITS_BASER3-entry-bytes	int	0x8	Number of bytes required per entry for GITS_BASER3 register.
GITS_BASER3-indirect-RAZ	bool	0x0	Indirect field for GITS_BASER3 register is RAZ/WI.
GITS_BASER3-type	int	0x0	Type field for GITS_BASER3 register. 0 = Unimplemented; 1 = Devices; 2 = Virtual Processors; 3 = Physical Processors; 4 = Collections
GITS_BASER4-entry-bytes	int	0x8	Number of bytes required per entry for GITS_BASER4 register.
GITS_BASER4-indirect-RAZ	bool	0x0	Indirect field for GITS_BASER4 register is RAZ/WI.
GITS_BASER4-type	int	0x0	Type field for GITS_BASER4 register. 0 = Unimplemented; 1 = Devices; 2 = Virtual Processors; 3 = Physical Processors; 4 = Collections
GITS_BASER5-entry-bytes	int	0x8	Number of bytes required per entry for GITS_BASER5 register.
GITS_BASER5-indirect-RAZ	bool	0x0	Indirect field for GITS_BASER5 register is RAZ/WI.
GITS_BASER5-type	int	0x0	Type field for GITS_BASER5 register. 0 = Unimplemented; 1 = Devices; 2 = Virtual Processors; 3 = Physical Processors; 4 = Collections
GITS_BASER6-entry-bytes	int	0x8	Number of bytes required per entry for GITS_BASER6 register.
GITS_BASER6-indirect-RAZ	bool	0x0	Indirect field for GITS_BASER6 register is RAZ/WI.
GITS_BASER6-type	int	0x0	Type field for GITS_BASER6 register. 0 = Unimplemented; 1 = Devices; 2 = Virtual Processors; 3 = Physical Processors; 4 = Collections
GITS_BASER7-entry-bytes	int	0x8	Number of bytes required per entry for GITS_BASER7 register.
GITS_BASER7-indirect-RAZ	bool	0x0	Indirect field for GITS_BASER7 register is RAZ/WI.
GITS_BASER7-type	int	0x0	Type field for GITS_BASER7 register. 0 = Unimplemented; 1 = Devices; 2 = Virtual Processors; 3 = Physical Processors; 4 = Collections
GITS_PIDR	int	0x0	The value for the GITS_PIDR registers, if non-zero. Note: fixed fields (device type etc.) will be overriden in this value.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-372 Parameters for GIC_IRI (continued)

Name	Type	Default value	Description
ICFGR-PPI-mask	int	0xaaaaaaaa	Mask for writes to ICFGR registers that configure PPIs
ICFGR-PPI-reset	int	0x0	Reset value for ICFGR regesters that configure PPIs
ICFGR-SGI-mask	int	0x0	Mask for writes to ICFGR registers that configure SGIs
ICFGR-SGI-reset	int	0xaaaaaaaa	Reset value for ICFGR registers that configure SGIs
ICFGR-SPI-mask	int	0xaaaaaaaa	Mask for writes to ICFGR registers that configure SPIs
ICFGR-SPI-reset	int	0x0	Reset value for ICFGR regesters that configure SPIs
ICFGR-rsvd-bit	bool	0x0	If ARE=0, the value of reserved bits i.e. bit 0,2,4..30 of ICFGRn for n>0
IGROUP-PPI-mask	int	0xffff	Mask for writes to PPI bits in IGROUP registers
IGROUP-PPI-reset	int	0x0	Reset value for SGI bits in IGROUP registers
IGROUP-SGI-mask	int	0xffff	Mask for writes to SGI bits in IGROUP registers
IGROUP-SGI-reset	int	0x0	Reset value for SGI bits in IGROUP registers
IIDR	int	0x0	GICD_IIDR and GICR_IIDR value
IRI-ID-bits	int	0x10	Number of bits used to represent interrupts IDs in the Distributor and Redistributors, forced to 10 when none of LPIs, extended SPIs or extended PPIs is supported
IROUTER-IRM-RAZ-WI	bool	0x0	GICD_IROUTERn.InterruptRoutingMode is RAZ/WI
ITS-BASER-force-page-alignement	bool	0x1	Force alignement of address writen to a GITS_BASER register to the page size configured
ITS-ID-bits	int	0x10	Number of interrupt bits supported by ITS.
ITS-MOVALL-update-collections	bool	0x0	Whether MOVALL command updates the collection entires
ITS-TRANSLATE64R	bool	0x0	Add an implementation specific register at 0x10008 supporting 64 bit TRANSLATER (dev[63:32], interupt[31:0])
ITS-collection-ID-bits	int	0x0	Number of collection bits supported by ITS (optional parameter, 0 => 16bits support and GITS_TYPER.CIL=0
ITS-count	int	0x0	Number of Interrupt Translation Services to be instantiated (0=none)
ITS-cumulative-collection-tables	bool	0x1	When true, the supported amount of collections is the sum of GITS_TYPER.HCC and the number of collections supported in memory, otherwise, simply the number supported in memory only. Irrelevant when HCC=0
ITS-device-bits	int	0x10	Number of bits supported for ITS device IDs.
ITS-entry-size	int	0x8	Number of bytes required to store each entry in the ITT tables.
ITS-hardware-collection-count	int	0x0	Number of hardware collections held exclusively in the ITS
ITS-legacy-iidr-typer-offset	bool	0x0	Put the GITS_IIDR and GITS_TYPER registers at their older offset of 0x8 and 0x4 respectively
ITS-shared-vPE-table	int	0x0	Number of affinity levels to which the vPE configuration table is shared. This parameter is valid when has-gicv4.1 is true.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-372 Parameters for GIC_IRI (continued)

Name	Type	Default value	Description
ITS-threaded-command-queue	bool	0x1	Enable execution of ITS commands in a separate thread which is sometimes required for cosimulation
ITS-use-physical-target-addresses	bool	0x1	Use physical hardware adresses for targets in ITS commands -- must be true for distributed implementations
ITS-vmovp-bit	bool	0x0	Device supports software issuing a VMOVP to only one of the ITSs that has a mapping for a vPE. The device itself ensures synchronization of the VMOVP command across all ITSs that have mapping for that vPE.
ITS0-base	int	0x0	Register base address for ITS0 (automatic if 0).
ITS1-base	int	0x0	Register base address for ITS1 (automatic if 0).
ITS2-base	int	0x0	Register base address for ITS2 (automatic if 0).
ITS3-base	int	0x0	Register base address for ITS3 (automatic if 0).
LPI-cache-check-data	bool	0x0	Enable Cached LPI data against memory checking when available for cache type
LPI-cache-type	int	0x1	Cache type for LPIs, 0:No caching, 1:Full caching
MSI_IIDR	int	0x0	Value returned in MSI_IIDR registers.
MSI_NS-frame0-base	int	0x0	If non-zero, sets the base address used for non-secure MSI frame 0 registers.
MSI_NS-frame0-max-SPI	int	0x0	Maximum SPI ID supported by non-secure MSI frame 0. Set to 0 to disable frame.
MSI_NS-frame0-min-SPI	int	0x0	Minimum SPI ID supported by non-secure MSI frame 0. Set to 0 to disable frame.
MSI_NS-frame1-base	int	0x0	If non-zero, sets the base address used for non-secure MSI frame 1 registers.
MSI_NS-frame1-max-SPI	int	0x0	Maximum SPI ID supported by non-secure MSI frame 1. Set to 0 to disable frame.
MSI_NS-frame1-min-SPI	int	0x0	Minimum SPI ID supported by non-secure MSI frame 1. Set to 0 to disable frame.
MSI_NS-frame2-base	int	0x0	If non-zero, sets the base address used for non-secure MSI frame 2 registers.
MSI_NS-frame2-max-SPI	int	0x0	Maximum SPI ID supported by non-secure MSI frame 2. Set to 0 to disable frame.
MSI_NS-frame2-min-SPI	int	0x0	Minimum SPI ID supported by non-secure MSI frame 2. Set to 0 to disable frame.
MSI_NS-frame3-base	int	0x0	If non-zero, sets the base address used for non-secure MSI frame 3 registers.
MSI_NS-frame3-max-SPI	int	0x0	Maximum SPI ID supported by non-secure MSI frame 3. Set to 0 to disable frame.
MSI_NS-frame3-min-SPI	int	0x0	Minimum SPI ID supported by non-secure MSI frame 3. Set to 0 to disable frame.
MSI_NS-frame4-base	int	0x0	If non-zero, sets the base address used for non-secure MSI frame 4 registers.
MSI_NS-frame4-max-SPI	int	0x0	Maximum SPI ID supported by non-secure MSI frame 4. Set to 0 to disable frame.
MSI_NS-frame4-min-SPI	int	0x0	Minimum SPI ID supported by non-secure MSI frame 4. Set to 0 to disable frame.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-372 Parameters for GIC_IRI (continued)

Name	Type	Default value	Description
MSI_NS-frame5-base	int	0x0	If non-zero, sets the base address used for non-secure MSI frame 5 registers.
MSI_NS-frame5-max-SPI	int	0x0	Maximum SPI ID supported by non-secure MSI frame 5. Set to 0 to disable frame.
MSI_NS-frame5-min-SPI	int	0x0	Minimum SPI ID supported by non-secure MSI frame 5. Set to 0 to disable frame.
MSI_NS-frame6-base	int	0x0	If non-zero, sets the base address used for non-secure MSI frame 6 registers.
MSI_NS-frame6-max-SPI	int	0x0	Maximum SPI ID supported by non-secure MSI frame 6. Set to 0 to disable frame.
MSI_NS-frame6-min-SPI	int	0x0	Minimum SPI ID supported by non-secure MSI frame 6. Set to 0 to disable frame.
MSI_NS-frame7-base	int	0x0	If non-zero, sets the base address used for non-secure MSI frame 7 registers.
MSI_NS-frame7-max-SPI	int	0x0	Maximum SPI ID supported by non-secure MSI frame 7. Set to 0 to disable frame.
MSI_NS-frame7-min-SPI	int	0x0	Minimum SPI ID supported by non-secure MSI frame 7. Set to 0 to disable frame.
MSI_PIDR	int	0x0	The value for the MSI_PIDR registers, if non-zero and distributor supports GICv2m. Note: fixed fields (device type etc.) will be overriden in this value.
MSI_S-frame0-base	int	0x0	If non-zero, sets the base address used for secure MSI frame 0 registers.
MSI_S-frame0-max-SPI	int	0x0	Maximum SPI ID supported by secure MSI frame 0. Set to 0 to disable frame.
MSI_S-frame0-min-SPI	int	0x0	Minimum SPI ID supported by secure MSI frame 0. Set to 0 to disable frame.
MSI_S-frame1-base	int	0x0	If non-zero, sets the base address used for secure MSI frame 1 registers.
MSI_S-frame1-max-SPI	int	0x0	Maximum SPI ID supported by secure MSI frame 1. Set to 0 to disable frame.
MSI_S-frame1-min-SPI	int	0x0	Minimum SPI ID supported by secure MSI frame 1. Set to 0 to disable frame.
MSI_S-frame2-base	int	0x0	If non-zero, sets the base address used for secure MSI frame 2 registers.
MSI_S-frame2-max-SPI	int	0x0	Maximum SPI ID supported by secure MSI frame 2. Set to 0 to disable frame.
MSI_S-frame2-min-SPI	int	0x0	Minimum SPI ID supported by secure MSI frame 2. Set to 0 to disable frame.
MSI_S-frame3-base	int	0x0	If non-zero, sets the base address used for secure MSI frame 3 registers.
MSI_S-frame3-max-SPI	int	0x0	Maximum SPI ID supported by secure MSI frame 3. Set to 0 to disable frame.
MSI_S-frame3-min-SPI	int	0x0	Minimum SPI ID supported by secure MSI frame 3. Set to 0 to disable frame.
MSI_S-frame4-base	int	0x0	If non-zero, sets the base address used for secure MSI frame 4 registers.
MSI_S-frame4-max-SPI	int	0x0	Maximum SPI ID supported by secure MSI frame 4. Set to 0 to disable frame.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-372 Parameters for GIC_IRI (continued)

Name	Type	Default value	Description
MSI_S-frame4-min-SPI	int	0x0	Minimum SPI ID supported by secure MSI frame 4. Set to 0 to disable frame.
MSI_S-frame5-base	int	0x0	If non-zero, sets the base address used for secure MSI frame 5 registers.
MSI_S-frame5-max-SPI	int	0x0	Maximum SPI ID supported by secure MSI frame 5. Set to 0 to disable frame.
MSI_S-frame5-min-SPI	int	0x0	Minimum SPI ID supported by secure MSI frame 5. Set to 0 to disable frame.
MSI_S-frame6-base	int	0x0	If non-zero, sets the base address used for secure MSI frame 6 registers.
MSI_S-frame6-max-SPI	int	0x0	Maximum SPI ID supported by secure MSI frame 6. Set to 0 to disable frame.
MSI_S-frame6-min-SPI	int	0x0	Minimum SPI ID supported by secure MSI frame 6. Set to 0 to disable frame.
MSI_S-frame7-base	int	0x0	If non-zero, sets the base address used for secure MSI frame 7 registers.
MSI_S-frame7-max-SPI	int	0x0	Maximum SPI ID supported by secure MSI frame 7. Set to 0 to disable frame.
MSI_S-frame7-min-SPI	int	0x0	Minimum SPI ID supported by secure MSI frame 7. Set to 0 to disable frame.
PA_SIZE	int	0x30	Number of valid bits in physical address
PPI-implemented-mask	int	0xffff	Mask of PPIs that are implemented. One bit per PPI bit 0 == PPI 16 (first PPI). This will affect other masks.
SPI-count	int	0xe0	Number of SPIs that are implemented.
SPI-message-based-support	bool	0x1	Distributor supports meassage based signaling of SPI
SPI-unimplemented	string	""	A comma spearated list of unimplemented SPIs ranges for sparse SPI defintion(for ex: '35, 39-42, 73)'
STATUSR-implemented	bool	0x1	Determines whether the GICR_STATUSR register is implemented.
allow-LPIEN-clear	bool	0x0	Allow RW behaviour on GICR_CTLR.LPIEN isntead of set once
chip-id	int	0x0	Chip ID when multichip operation is enabled.
common-lpi-configuration	int	0x0	Describes which re-distributors share (and must be configured with the same) LPI configuration table as described in GICR_TYPER(0:All, 1:A.x.x.x, 2:A.B.x.x, 3:A.B.C.x
common-vPE-table-affinity	string	""	Affinity value list in the form of 'a.b.c.d, e.f.g.h, etc' given to the ITS(s) (where a.b.c.d corresponds to ITS0, e.f.g.h corresponds to ITS1 and so on). Under an affinity value the vPE configuration table is shared among redistributors where the level to be shared is defined by ITS-shared-vPE-table. This parameter is valid when has-gicv4.1 is true.
consolidators	string	""	Specify consolidators' base addresses, interrupt line counts and base interrupt IDs, in the form 'baseAddr0:itlineCount0:baseINTID0, baseAddr1:itlineCount1:baseINTID1, [etc]' (eg '0x3f100000:64:4096, 0x3f200000:64:4224'). The consolidators' count is inferred from the list (maximum of 4). If not specified, the component contains no consolidators.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-372 Parameters for GIC_IRI (continued)

Name	Type	Default value	Description
delay-ITS-accesses	bool	0x1	Delay accesses from the ITS until GICR_SYNCR is read.
delay-redistributor-accesses	bool	0x1	Delay memory accesses from the redistributor until GICR_SYNCR is read.
direct-lpi-support	bool	0x0	Enable support for LPI operations through GICR registers
enable-multichip-operation	bool	0x0	Enables multi-chip operation between Distributors in distributed GIC IRI
enable_protocol_checking	bool	0x0	Enable/disable protocol checking at cpu interface
enabled	bool	0x1	Enable GICv3 functionality; when false the component is inactive.
extended-ppi-count	int	0x0	Number of extended PPI supported
extended-spi-count	int	0x0	Number of extended SPI supported
fixed-routed-spis	string	""	Value of IROUTER[n] register in the form 'n=a.b.c.d, n=*'. The RM bit of IROUTER is 0 when n=a.b.c.d is used else 1 when n=* is used. n can be >= 32 and <= 1019
gicv2-only	bool	0x0	If true, when using the GICv3 model, pretend to be a GICv2 system
has-two-security-states	bool	0x1	If true, has two security states
has_VPENDBASER-dirty-flag-on-load	bool	0x0	GICR_VPENDBASER.Dirty reflects transient loading state when valid=1
has_mpam	bool	0x0	Enable MPAM support on ITS and RDs
ignore-generate-sgi-when-no-are	bool	0x0	Ignore GenerateSGI packets coming form the CPU interface if both ARE_S and ARE_NS are 0
irouter-default-mask	string	""	Default Mask value for IROUTER[32..1019] register in the form 'a.b.c.d'
irouter-default-reset	string	""	Default Reset Value of IROUTER[32..1019] register in the form 'a.b.c.d' or *
irouter-mask-values	string	""	Mask Value of IROUTER[n] register in the form 'n=a.b.c.d'.n can be >= 32 and <= 1019
irouter-reset-values	string	""	Reset Value of IROUTER[n] register in the form 'n=a.b.c.d or n=*'.n can be >= 32 and <= 1019
legacy-sgi-enable-rao	bool	0x0	Enables for SGI associated with an ARE=0 regime are RAO/WI
local-SEIs	bool	0x0	Generate SEI to signal internal issues
local-VSEIs	bool	0x0	Generate VSEI to signal internal issues
lockable-SPI-count	int	0x0	Number of SPIs that are locked down when CFGSDISABLE signal is asserted. Only applies for GICv2.
max-pe-on-chip	int	0x4	Maximum number of cores on any single chip. This will be used to identify the target chip and core when multichip operation is enabled.
monolithic	bool	0x0	Indicate that the implementation is not distributed
mpam_max_partid	int	0xffff	Maximum valid PARTID
mpam_max_pmg	int	0xff	Maximum valid PMG

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-372 Parameters for GIC_IRI (continued)

Name	Type	Default value	Description
multichip-threaded-dgi	bool	0x1	Enable sending multichip DGI messages in a separate thread. This parameter is valid only if enable-multichip-operation is set.
non-ARE-core-count	int	0x8	Maximum number of non-ARE cores; normally used to pass the cluster-level NUM_CORES parameter to the top-level redistributor.
outer-cacheability-support	bool	0x0	Allow configuration of outer cachability attributes in ITS and Redistributor
output_attributes	string	"ExtendedID[62:55]=MPAM_PMG, ExtendedID[54:39]=MPAM_PARTID, ExtendedID[38]=MPAM_NS"	User-defined transform to be applied to bus attributes like MasterID, ExtendedID or UserFlags. Currently, only works for MPAM Attributes encoding into bus attributes.
print-memory-map	bool	0x0	Print memory map to stdout
priority-bits	int	0x5	Number of implemented priority bits
processor-numbers	string	""	Specify processor numbers (as appears in GICR_TYPER) in the form 0.0.0.0=0,0.0.0.1=1 etc.) If not specified, will number processors starting at 0.
redistributor-threaded-sync	bool	0x1	Enable execution of redistributor delayed transactions in a separate thread which is sometimes required for cosimulation
reg-base	int	0x2c010000	Base for decoding GICv3 registers.
reg-base-per-redistributor	string	""	Base address for each redistributor in the form '0.0.0.0=0x2c010000, 0.0.0.1=0x2c020000'. All redistributors must be specified and this overrides the reg-base parameter (except that reg-base will still be used for the top-level redistributor). If reg-base-per-redistributor-file is specified, this parameter is ignored.
reg-base-per-redistributor-file	string	""	Path to file containing the base address for each redistributor in the form '0.0.0.0=0x2c010000, 0.0.0.1=0x2c020000'. All redistributors must be specified and this overrides the reg-base parameter (except that reg-base will still be used for the top-level redistributor). If this parameter is specified, reg-base-per-redistributor parameter will be ignored even when it is given.
report-MSI-error-via-statusr	int	0x0	Report MSI error via GITS_STATUSR. (0:unsupported, 1:report by GITS_STATUSR, 2:report by GITS_STATUSR and interrupt as well).
sgi-range-selector-support	bool	0x0	Device has support for the Range Selector feature for SGI
single-set-support	bool	0x0	When true, forces redistributors to recall interrupts with a clear rather than issue a second Set command
supports-shareability	bool	0x1	Device supports shareability attributes on outgoing memory bus (i.e. is modelling an ACElite port rather than an AXI4 port).
trace-speculative-lpi-property-update	bool	0x0	Trace LPI propery updates performed on speculative accesses (useful for debuging LPI)
virtual-lpi-support	bool	0x0	GICv4 Virtual LPIs and Direct injection of Virtual LPIs supported

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-372 Parameters for GIC_IRI (continued)

Name	Type	Default value	Description
`virtual-priority-bits`	`int`	`0x5`	Number of implemented virtual priority bits
`wakeup-on-reset`	`bool`	`0x0`	Go against specification and start redistributors in woken-up state at reset. This allows software that was written for previous versions of the GICv3 specification to work correctly. This should not be used for production code or when the distributor is used separately from the core fast model.

3.9.29 GICv3IRI

GICv3 metacomponent for redistribution of interrupts(contains Distributor, and configurable numbers of ITSs and Redistributors. This model is written in C++.

GICv3IRI contains the following CADI targets:

- GICv3IRI

GICv3IRI contains the following MTI components:

- *4.76 GICv3Distributor - trace* on page 4-2496
- *4.77 GICv3IRI - trace* on page 4-2504
- *4.80 GICv3Redistributor - trace* on page 4-2527
- *4.81 GICv3RedistributorInternal - trace* on page 4-2541

About GICv3IRI

The GICv3IRI has one slave PVBus interface and one master PVBus interface. It behaves in a similar manner to a GICv3-compatible device, with the slave interface, `pvbus_s`, granting access to the register banks used by the configuration and operation of MSIs and the master interface, `pvbus_m`, issuing transactions that are required by the ITS and the redistributors in LPI-related operations. All transactions that are routed to the slave port terminate in the component. Accesses to unmapped space are RAZ/WI.

An instance of GICv3 requires a small set of parameters to be configured to be useful. For example:

```
gic_iri : GICv3IRI(
    "reg-base" = 0xF0020000, //Base address for GICD_* REGISTERS, 64K aligned
    "CPU-affinities" = "0.0.0.0, 0.0.1.0, 0.0.1.1",
        //A comma-separated list of affinity addresses corresponding to
        //cpu affinities in the system
    "reg-base-per-redistributor"="0.0.0.0=0xF0040000,0.0.1.0=0xF0060000, 0.0.0.0=0xF0080000",
        //Base addresses for each redistributor in a comma-separated list of
        //affinity=address
);
```

To use LPIs, an ITS must be configured. A minimal configuration might consist of, for example:

```
    "ITS-count" = 1, //The number of ITSs in the IRI. Defaults to zero.
    "ITS0-base" = 0xF0100000,
    "GITS_BASER0-type" = 1, //Type 1 is Devices. A device table is always needed.
    "GITS_BASER2-type" = 4, //Type 4 is Collections.
                            //A collection table is needed if GITS_TYPER.HCC is 0.
```

To use GICv4 functionality, one or more ITSs must be configured, as shown in the previous example. In addition, the following parameters are required:

```
    "virtual-lpi-support"=true,
    "GITS_BASER4-type"=2 //Type 2 is Virtual PEs.
                         //Such a table is needed for GICv4 functionality.
```


Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Note

- Set the `FASTSIM_GIC_MEMORY_MAP` environment variable to 1 to print to `stderr` the memory map of any GICv3 or later models that are included in the platform being run.
- Set the `GICD_ITARGETSR-RAZWI` parameter to true for legacy, GICv2-style routing, where interrupts are routed to the first processor in the system.

Table 3-373 Ports

Name	Protocol	Type	Description
`cfgsdisable`	*Signal* on page 2-85	Slave	Disable some SPIs signal.
`extended_ppi_in_n[64]`	*Signal* on page 2-85	Slave	Extended private peripheral interrupts (ID1056-ID1119) for cpu *n*, where *n* is in the range 0-255.
`extended_spi_in[1024]`	*Signal* on page 2-85	Slave	Extended Shared peripheral interrupts.
`msi_error_interrupt`	*Signal* on page 2-85	Master	When report of MSI error allowed through interrupt, loop-back this signal to relevant IRQ input signal
`po_reset`	*Signal* on page 2-85	Slave	Resets.
`ppi_in_n[16]`	*Signal* on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu *n*, where *n* is in the range 0-255.
`pvbus_m`	`PVBus`	Master	Memory bus out: transactions generated by the IRI.
`pvbus_s`	`PVBus`	Slave	Memory bus in: memory-mapped register accesses are accepted on this interface.
`redistributor_m[256]`	*GICv3Comms* on page 2-84	Master	Input from and output to CPU interface.
`reset`	*Signal* on page 2-85	Slave	Resets.
`spi_in[988]`	*Signal* on page 2-85	Slave	Shared peripheral interrupts.
`wake_request[256]`	*Signal* on page 2-85	Master	Power management outputs.
`wire_to_msi_0[1024]`	*Signal* on page 2-85	Slave	Wire-to-MSI interrupts for architectural consolidator 0.
`wire_to_msi_1[1024]`	*Signal* on page 2-85	Slave	Wire-to-MSI interrupts for architectural consolidator 1.
`wire_to_msi_2[1024]`	*Signal* on page 2-85	Slave	Wire-to-MSI interrupts for architectural consolidator 2.
`wire_to_msi_3[1024]`	*Signal* on page 2-85	Slave	Wire-to-MSI interrupts for architectural consolidator 3.

Table 3-374 Parameters for GICv3IRI

Name	Type	Default value	Description
`A3-affinity-supported`	`bool`	`0x0`	Device supports affinity level 3 values that are non-zero.
`ARE-fixed-to-one`	`bool`	`0x0`	GICv2 compatibility is not supported and GICD_CTLR.ARE_* is always one
`CPU-affinities`	`string`	`""`	A comma separated list of dotted quads containing the affinities of all PEs connected to this IRI. If CPU-affinities-file is specified, this parameter is ignored.
`CPU-affinities-file`	`string`	`""`	A file containing comma separated list of dotted quads containing the affinities of all PEs connected to this IRI. If this parameter is specified, CPU-affinities parameter will be ignored even when it is given.
`DPG-ARE-only`	`bool`	`0x0`	Limit application of DPG bits to interrupt groups for which ARE=1

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-374 Parameters for GICv3IRI (continued)

Name	Type	Default value	Description
DPG-bits-implemented	bool	0x0	Enable implementation of interrupt group participation bits or DPG bits in GICR_CTLR
DS-fixed-to-zero	bool	0x0	Enable/disable support of single security state
GICD-alias	int	0x0	In GICv2 mode: the base address for a 4k page alias of the first 4k of the Distributor page, in GICv3 mode. the base address of a 64KB page containing message based SPI signalling register aliases(0:Disabled)
GICD-legacy-registers-as-reserved	bool	0x0	When ARE is RAO/WI, makes superfluous registers in GICD reserved (including for the purpose of STATUSR updates)
GICD_CTLR-DS-1-means-secure-only	bool	0x0	If GICD_CTLR.DS=1, GICD supports a single security state which is secure if this is true, otherwise is non-secure.
GICD_ITARGETSR-RAZWI	bool	0x0	If true, the GICD_ITARGETS registers are RAZ/WI
GICD_PIDR	int	0x0	The value for the GICD_PIDR registers, if non-zero. Note: fixed fields (device type etc.) will be overriden in this value.
GICR-clear-enable-supported	bool	0x0	When true, this sets the value of the RO bit GICR_CTLR.CES with the value of the parameter allow-LPIEN-clear, making it visible to software
GICR-invalidate-registers-implemented	bool	0x0	When true, the registers GICR_INVLPIR, GICR_INVALLR and GICR_SYNCR are implemented
GICR_PIDR	int	0x0	The value for the GICR_PIDR registers, if non-zero. Note: fixed fields (device type etc.) will be overriden in this value.
GICR_PROPBASER-read-only	bool	0x0	GICR_PROPBASER register is read-only.
GICR_PROPBASER-reset-value	int	0x0	Value of GICR_PROPBASER on reset.
GITS_BASER0-entry-bytes	int	0x8	Number of bytes required per entry for GITS_BASER0 register.
GITS_BASER0-indirect-RAZ	bool	0x0	Indirect field for GITS_BASER0 register is RAZ/WI.
GITS_BASER0-type	int	0x0	Type field for GITS_BASER0 register. 0 = Unimplemented; 1 = Devices; 2 = Virtual Processors; 3 = Physical Processors; 4 = Collections
GITS_BASER1-entry-bytes	int	0x8	Number of bytes required per entry for GITS_BASER1 register.
GITS_BASER1-indirect-RAZ	bool	0x0	Indirect field for GITS_BASER1 register is RAZ/WI.
GITS_BASER1-type	int	0x0	Type field for GITS_BASER1 register. 0 = Unimplemented; 1 = Devices; 2 = Virtual Processors; 3 = Physical Processors; 4 = Collections
GITS_BASER2-entry-bytes	int	0x8	Number of bytes required per entry for GITS_BASER2 register.
GITS_BASER2-indirect-RAZ	bool	0x0	Indirect field for GITS_BASER2 register is RAZ/WI.
GITS_BASER2-type	int	0x0	Type field for GITS_BASER2 register. 0 = Unimplemented; 1 = Devices; 2 = Virtual Processors; 3 = Physical Processors; 4 = Collections

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-374 Parameters for GICv3IRI (continued)

Name	Type	Default value	Description
GITS_BASER3-entry-bytes	int	0x8	Number of bytes required per entry for GITS_BASER3 register.
GITS_BASER3-indirect-RAZ	bool	0x0	Indirect field for GITS_BASER3 register is RAZ/WI.
GITS_BASER3-type	int	0x0	Type field for GITS_BASER3 register. 0 = Unimplemented; 1 = Devices; 2 = Virtual Processors; 3 = Physical Processors; 4 = Collections
GITS_BASER4-entry-bytes	int	0x8	Number of bytes required per entry for GITS_BASER4 register.
GITS_BASER4-indirect-RAZ	bool	0x0	Indirect field for GITS_BASER4 register is RAZ/WI.
GITS_BASER4-type	int	0x0	Type field for GITS_BASER4 register. 0 = Unimplemented; 1 = Devices; 2 = Virtual Processors; 3 = Physical Processors; 4 = Collections
GITS_BASER5-entry-bytes	int	0x8	Number of bytes required per entry for GITS_BASER5 register.
GITS_BASER5-indirect-RAZ	bool	0x0	Indirect field for GITS_BASER5 register is RAZ/WI.
GITS_BASER5-type	int	0x0	Type field for GITS_BASER5 register. 0 = Unimplemented; 1 = Devices; 2 = Virtual Processors; 3 = Physical Processors; 4 = Collections
GITS_BASER6-entry-bytes	int	0x8	Number of bytes required per entry for GITS_BASER6 register.
GITS_BASER6-indirect-RAZ	bool	0x0	Indirect field for GITS_BASER6 register is RAZ/WI.
GITS_BASER6-type	int	0x0	Type field for GITS_BASER6 register. 0 = Unimplemented; 1 = Devices; 2 = Virtual Processors; 3 = Physical Processors; 4 = Collections
GITS_BASER7-entry-bytes	int	0x8	Number of bytes required per entry for GITS_BASER7 register.
GITS_BASER7-indirect-RAZ	bool	0x0	Indirect field for GITS_BASER7 register is RAZ/WI.
GITS_BASER7-type	int	0x0	Type field for GITS_BASER7 register. 0 = Unimplemented; 1 = Devices; 2 = Virtual Processors; 3 = Physical Processors; 4 = Collections
GITS_PIDR	int	0x0	The value for the GITS_PIDR registers, if non-zero. Note: fixed fields (device type etc.) will be overriden in this value.
ICFGR-PPI-mask	int	0xaaaaaaaa	Mask for writes to ICFGR registers that configure PPIs
ICFGR-PPI-reset	int	0x0	Reset value for ICFGR regesters that configure PPIs
ICFGR-SGI-mask	int	0x0	Mask for writes to ICFGR registers that configure SGIs
ICFGR-SGI-reset	int	0xaaaaaaaa	Reset value for ICFGR registers that configure SGIs
ICFGR-SPI-mask	int	0xaaaaaaaa	Mask for writes to ICFGR registers that configure SPIs
ICFGR-SPI-reset	int	0x0	Reset value for ICFGR regesters that configure SPIs
ICFGR-rsvd-bit	bool	0x0	If ARE=0, the value of reserved bits i.e. bit 0,2,4..30 of ICFGRn for n>0
IGROUP-PPI-mask	int	0xffff	Mask for writes to PPI bits in IGROUP registers

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-374 Parameters for GICv3IRI (continued)

Name	Type	Default value	Description
IGROUP-PPI-reset	int	0x0	Reset value for SGI bits in IGROUP registers
IGROUP-SGI-mask	int	0xffff	Mask for writes to SGI bits in IGROUP registers
IGROUP-SGI-reset	int	0x0	Reset value for SGI bits in IGROUP registers
IIDR	int	0x0	GICD_IIDR and GICR_IIDR value
IRI-ID-bits	int	0x10	Number of bits used to represent interrupts IDs in the Distributor and Redistributors, forced to 10 when none of LPIs, extended SPIs or extended PPIs is supported
IROUTER-IRM-RAZ-WI	bool	0x0	GICD_IROUTERn.InterruptRoutingMode is RAZ/WI
ITS-BASER-force-page-alignement	bool	0x1	Force alignement of address writen to a GITS_BASER register to the page size configured
ITS-ID-bits	int	0x10	Number of interrupt bits supported by ITS.
ITS-MOVALL-update-collections	bool	0x0	Whether MOVALL command updates the collection entires
ITS-TRANSLATE64R	bool	0x0	Add an implementation specific register at 0x10008 supporting 64 bit TRANSLATER (dev[63:32], interupt[31:0])
ITS-collection-ID-bits	int	0x0	Number of collection bits supported by ITS (optional parameter, 0 => 16bits support and GITS_TYPER.CIL=0
ITS-count	int	0x0	Number of Interrupt Translation Services to be instantiated (0=none)
ITS-cumulative-collection-tables	bool	0x1	When true, the supported amount of collections is the sum of GITS_TYPER.HCC and the number of collections supported in memory, otherwise, simply the number supported in memory only. Irrelevant when HCC=0
ITS-device-bits	int	0x10	Number of bits supported for ITS device IDs.
ITS-entry-size	int	0x8	Number of bytes required to store each entry in the ITT tables.
ITS-hardware-collection-count	int	0x0	Number of hardware collections held exclusively in the ITS
ITS-legacy-iidr-typer-offset	bool	0x0	Put the GITS_IIDR and GITS_TYPER registers at their older offset of 0x8 and 0x4 respectively
ITS-shared-vPE-table	int	0x0	Number of affinity levels to which the vPE configuration table is shared. This parameter is valid when has-gicv4.1 is true.
ITS-threaded-command-queue	bool	0x1	Enable execution of ITS commands in a separate thread which is sometimes required for cosimulation
ITS-use-physical-target-addresses	bool	0x1	Use physical hardware adresses for targets in ITS commands -- must be true for distributed implementations
ITS-vmovp-bit	bool	0x0	Device supports software issuing a VMOVP to only one of the ITSs that has a mapping for a vPE. The device itself ensures synchronization of the VMOVP command across all ITSs that have mapping for that vPE.
ITS0-base	int	0x0	Register base address for ITS0 (automatic if 0).
ITS1-base	int	0x0	Register base address for ITS1 (automatic if 0).
ITS2-base	int	0x0	Register base address for ITS2 (automatic if 0).

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-374 Parameters for GICv3IRI (continued)

Name	Type	Default value	Description
ITS3-base	int	0x0	Register base address for ITS3 (automatic if 0).
LPI-cache-check-data	bool	0x0	Enable Cached LPI data against memory checking when available for cache type
LPI-cache-type	int	0x1	Cache type for LPIs, 0:No caching, 1:Full caching
MSI_IIDR	int	0x0	Value returned in MSI_IIDR registers.
MSI_NS-frame0-base	int	0x0	If non-zero, sets the base address used for non-secure MSI frame 0 registers.
MSI_NS-frame0-max-SPI	int	0x0	Maximum SPI ID supported by non-secure MSI frame 0. Set to 0 to disable frame.
MSI_NS-frame0-min-SPI	int	0x0	Minimum SPI ID supported by non-secure MSI frame 0. Set to 0 to disable frame.
MSI_NS-frame1-base	int	0x0	If non-zero, sets the base address used for non-secure MSI frame 1 registers.
MSI_NS-frame1-max-SPI	int	0x0	Maximum SPI ID supported by non-secure MSI frame 1. Set to 0 to disable frame.
MSI_NS-frame1-min-SPI	int	0x0	Minimum SPI ID supported by non-secure MSI frame 1. Set to 0 to disable frame.
MSI_NS-frame2-base	int	0x0	If non-zero, sets the base address used for non-secure MSI frame 2 registers.
MSI_NS-frame2-max-SPI	int	0x0	Maximum SPI ID supported by non-secure MSI frame 2. Set to 0 to disable frame.
MSI_NS-frame2-min-SPI	int	0x0	Minimum SPI ID supported by non-secure MSI frame 2. Set to 0 to disable frame.
MSI_NS-frame3-base	int	0x0	If non-zero, sets the base address used for non-secure MSI frame 3 registers.
MSI_NS-frame3-max-SPI	int	0x0	Maximum SPI ID supported by non-secure MSI frame 3. Set to 0 to disable frame.
MSI_NS-frame3-min-SPI	int	0x0	Minimum SPI ID supported by non-secure MSI frame 3. Set to 0 to disable frame.
MSI_NS-frame4-base	int	0x0	If non-zero, sets the base address used for non-secure MSI frame 4 registers.
MSI_NS-frame4-max-SPI	int	0x0	Maximum SPI ID supported by non-secure MSI frame 4. Set to 0 to disable frame.
MSI_NS-frame4-min-SPI	int	0x0	Minimum SPI ID supported by non-secure MSI frame 4. Set to 0 to disable frame.
MSI_NS-frame5-base	int	0x0	If non-zero, sets the base address used for non-secure MSI frame 5 registers.
MSI_NS-frame5-max-SPI	int	0x0	Maximum SPI ID supported by non-secure MSI frame 5. Set to 0 to disable frame.
MSI_NS-frame5-min-SPI	int	0x0	Minimum SPI ID supported by non-secure MSI frame 5. Set to 0 to disable frame.
MSI_NS-frame6-base	int	0x0	If non-zero, sets the base address used for non-secure MSI frame 6 registers.
MSI_NS-frame6-max-SPI	int	0x0	Maximum SPI ID supported by non-secure MSI frame 6. Set to 0 to disable frame.
MSI_NS-frame6-min-SPI	int	0x0	Minimum SPI ID supported by non-secure MSI frame 6. Set to 0 to disable frame.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-374 Parameters for GICv3IRI (continued)

Name	Type	Default value	Description
MSI_NS-frame7-base	int	0x0	If non-zero, sets the base address used for non-secure MSI frame 7 registers.
MSI_NS-frame7-max-SPI	int	0x0	Maximum SPI ID supported by non-secure MSI frame 7. Set to 0 to disable frame.
MSI_NS-frame7-min-SPI	int	0x0	Minimum SPI ID supported by non-secure MSI frame 7. Set to 0 to disable frame.
MSI_PIDR	int	0x0	The value for the MSI_PIDR registers, if non-zero and distributor supports GICv2m. Note: fixed fields (device type etc.) will be overriden in this value.
MSI_S-frame0-base	int	0x0	If non-zero, sets the base address used for secure MSI frame 0 registers.
MSI_S-frame0-max-SPI	int	0x0	Maximum SPI ID supported by secure MSI frame 0. Set to 0 to disable frame.
MSI_S-frame0-min-SPI	int	0x0	Minimum SPI ID supported by secure MSI frame 0. Set to 0 to disable frame.
MSI_S-frame1-base	int	0x0	If non-zero, sets the base address used for secure MSI frame 1 registers.
MSI_S-frame1-max-SPI	int	0x0	Maximum SPI ID supported by secure MSI frame 1. Set to 0 to disable frame.
MSI_S-frame1-min-SPI	int	0x0	Minimum SPI ID supported by secure MSI frame 1. Set to 0 to disable frame.
MSI_S-frame2-base	int	0x0	If non-zero, sets the base address used for secure MSI frame 2 registers.
MSI_S-frame2-max-SPI	int	0x0	Maximum SPI ID supported by secure MSI frame 2. Set to 0 to disable frame.
MSI_S-frame2-min-SPI	int	0x0	Minimum SPI ID supported by secure MSI frame 2. Set to 0 to disable frame.
MSI_S-frame3-base	int	0x0	If non-zero, sets the base address used for secure MSI frame 3 registers.
MSI_S-frame3-max-SPI	int	0x0	Maximum SPI ID supported by secure MSI frame 3. Set to 0 to disable frame.
MSI_S-frame3-min-SPI	int	0x0	Minimum SPI ID supported by secure MSI frame 3. Set to 0 to disable frame.
MSI_S-frame4-base	int	0x0	If non-zero, sets the base address used for secure MSI frame 4 registers.
MSI_S-frame4-max-SPI	int	0x0	Maximum SPI ID supported by secure MSI frame 4. Set to 0 to disable frame.
MSI_S-frame4-min-SPI	int	0x0	Minimum SPI ID supported by secure MSI frame 4. Set to 0 to disable frame.
MSI_S-frame5-base	int	0x0	If non-zero, sets the base address used for secure MSI frame 5 registers.
MSI_S-frame5-max-SPI	int	0x0	Maximum SPI ID supported by secure MSI frame 5. Set to 0 to disable frame.
MSI_S-frame5-min-SPI	int	0x0	Minimum SPI ID supported by secure MSI frame 5. Set to 0 to disable frame.
MSI_S-frame6-base	int	0x0	If non-zero, sets the base address used for secure MSI frame 6 registers.
MSI_S-frame6-max-SPI	int	0x0	Maximum SPI ID supported by secure MSI frame 6. Set to 0 to disable frame.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-374 Parameters for GICv3IRI (continued)

Name	Type	Default value	Description
MSI_S-frame6-min-SPI	int	0x0	Minimum SPI ID supported by secure MSI frame 6. Set to 0 to disable frame.
MSI_S-frame7-base	int	0x0	If non-zero, sets the base address used for secure MSI frame 7 registers.
MSI_S-frame7-max-SPI	int	0x0	Maximum SPI ID supported by secure MSI frame 7. Set to 0 to disable frame.
MSI_S-frame7-min-SPI	int	0x0	Minimum SPI ID supported by secure MSI frame 7. Set to 0 to disable frame.
PA_SIZE	int	0x30	Number of valid bits in physical address
PPI-implemented-mask	int	0xffff	Mask of PPIs that are implemented. One bit per PPI bit 0 == PPI 16 (first PPI). This will affect other masks.
SPI-count	int	0xe0	Number of SPIs that are implemented.
SPI-message-based-support	bool	0x1	Distributor supports meassage based signaling of SPI
SPI-unimplemented	string	""	A comma spearated list of unimplemented SPIs ranges for sparse SPI defintion(for ex: '35, 39-42, 73)'
STATUSR-implemented	bool	0x1	Determines whether the GICR_STATUSR register is implemented.
allow-LPIEN-clear	bool	0x0	Allow RW behaviour on GICR_CTLR.LPIEN isntead of set once
chip-id	int	0x0	Chip ID when multichip operation is enabled.
common-lpi-configuration	int	0x0	Describes which re-distributors share (and must be configured with the same) LPI configuration table as described in GICR_TYPER(0:All, 1:A.x.x.x, 2:A.B.x.x, 3:A.B.C.x
common-vPE-table-affinity	string	""	Affinity value list in the form of 'a.b.c.d, e.f.g.h, etc' given to the ITS(s) (where a.b.c.d corresponds to ITS0, e.f.g.h corresponds to ITS1 and so on). Under an affinity value the vPE configuration table is shared among redistributors where the level to be shared is defined by ITS-shared-vPE-table. This parameter is valid when has-gicv4.1 is true.
consolidators	string	""	Specify consolidators' base addresses, interrupt line counts and base interrupt IDs, in the form 'baseAddr0:itlineCount0:baseINTID0, baseAddr1:itlineCount1:baseINTID1, [etc]' (eg '0x3f100000:64:4096, 0x3f200000:64:4224'). The consolidators' count is inferred from the list (maximum of 4). If not specified, the component contains no consolidators.
delay-ITS-accesses	bool	0x1	Delay accesses from the ITS until GICR_SYNCR is read.
delay-redistributor-accesses	bool	0x1	Delay memory accesses from the redistributor until GICR_SYNCR is read.
direct-lpi-support	bool	0x0	Enable support for LPI operations through GICR registers
enable-multichip-operation	bool	0x0	Enables multi-chip operation between Distributors in distributed GIC IRI
enable_protocol_checking	bool	0x0	Enable/disable protocol checking at cpu interface
enabled	bool	0x1	Enable GICv3 functionality; when false the component is inactive.
extended-ppi-count	int	0x0	Number of extended PPI supported

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-374 Parameters for GICv3IRI (continued)

Name	Type	Default value	Description
extended-spi-count	int	0x0	Number of extended SPI supported
fixed-routed-spis	string	""	Value of IROUTER[n] register in the form 'n=a.b.c.d, n=*'. The RM bit of IROUTER is 0 when n=a.b.c.d is used else 1 when n=* is used. n can be >= 32 and <= 1019
gicv2-only	bool	0x0	If true, when using the GICv3 model, pretend to be a GICv2 system
has-two-security-states	bool	0x1	If true, has two security states
has_VPENDBASER-dirty-flag-on-load	bool	0x0	GICR_VPENDBASER.Dirty reflects transient loading state when valid=1
has_mpam	bool	0x0	Enable MPAM support on ITS and RDs
ignore-generate-sgi-when-no-are	bool	0x0	Ignore GenerateSGI packets coming form the CPU interface if both ARE_S and ARE_NS are 0
irouter-default-mask	string	""	Default Mask value for IROUTER[32..1019] register in the form 'a.b.c.d'
irouter-default-reset	string	""	Default Reset Value of IROUTER[32..1019] register in the form 'a.b.c.d' or *
irouter-mask-values	string	""	Mask Value of IROUTER[n] register in the form 'n=a.b.c.d'.n can be >= 32 and <= 1019
irouter-reset-values	string	""	Reset Value of IROUTER[n] register in the form 'n=a.b.c.d or n=*'.n can be >= 32 and <= 1019
legacy-sgi-enable-rao	bool	0x0	Enables for SGI associated with an ARE=0 regime are RAO/WI
local-SEIs	bool	0x0	Generate SEI to signal internal issues
local-VSEIs	bool	0x0	Generate VSEI to signal internal issues
lockable-SPI-count	int	0x0	Number of SPIs that are locked down when CFGSDISABLE signal is asserted. Only applies for GICv2.
max-pe-on-chip	int	0x4	Maximum number of cores on any single chip. This will be used to identify the target chip and core when multichip operation is enabled.
monolithic	bool	0x0	Indicate that the implementation is not distributed
mpam_max_partid	int	0xffff	Maximum valid PARTID
mpam_max_pmg	int	0xff	Maximum valid PMG
multichip-threaded-dgi	bool	0x1	Enable sending multichip DGI messages in a separate thread. This parameter is valid only if enable-multichip-operation is set.
non-ARE-core-count	int	0x8	Maximum number of non-ARE cores; normally used to pass the cluster-level NUM_CORES parameter to the top-level redistributor.
outer-cacheability-support	bool	0x0	Allow configuration of outer cachability attributes in ITS and Redistributor

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-374 Parameters for GICv3IRI (continued)

Name	Type	Default value	Description
output_attributes	string	"ExtendedID[62:55]=MPAM_PMG, ExtendedID[54:39]=MPAM_PARTID, ExtendedID[38]=MPAM_NS"	User-defined transform to be applied to bus attributes like MasterID, ExtendedID or UserFlags. Currently, only works for MPAM Attributes encoding into bus attributes.
print-memory-map	bool	0x0	Print memory map to stdout
priority-bits	int	0x5	Number of implemented priority bits
processor-numbers	string	""	Specify processor numbers (as appears in GICR_TYPER) in the form 0.0.0.0=0,0.0.0.1=1 etc.) If not specified, will number processors starting at 0.
redistributor-threaded-sync	bool	0x1	Enable execution of redistributor delayed transactions in a separate thread which is sometimes required for cosimulation
reg-base	int	0x2c010000	Base for decoding GICv3 registers.
reg-base-per-redistributor	string	""	Base address for each redistributor in the form '0.0.0.0=0x2c010000, 0.0.0.1=0x2c020000'. All redistributors must be specified and this overrides the reg-base parameter (except that reg-base will still be used for the top-level redistributor). If reg-base-per-redistributor-file is specified, this parameter is ignored.
reg-base-per-redistributor-file	string	""	Path to file containing the base address for each redistributor in the form '0.0.0.0=0x2c010000, 0.0.0.1=0x2c020000'. All redistributors must be specified and this overrides the reg-base parameter (except that reg-base will still be used for the top-level redistributor). If this parameter is specified, reg-base-per-redistributor parameter will be ignored even when it is given.
report-MSI-error-via-statusr	int	0x0	Report MSI error via GITS_STATUSR. (0:unsupported, 1:report by GITS_STATUSR, 2:report by GITS_STATUSR and interrupt as well).
sgi-range-selector-support	bool	0x0	Device has support for the Range Selector feature for SGI
single-set-support	bool	0x0	When true, forces redistributors to recall interrupts with a clear rather than issue a second Set command
supports-shareability	bool	0x1	Device supports shareability attributes on outgoing memory bus (i.e. is modelling an ACElite port rather than an AXI4 port).
trace-speculative-lpi-property-update	bool	0x0	Trace LPI propery updates performed on speculative accesses (useful for debuging LPI)
virtual-lpi-support	bool	0x0	GICv4 Virtual LPIs and Direct injection of Virtual LPIs supported
virtual-priority-bits	int	0x5	Number of implemented virtual priority bits
wakeup-on-reset	bool	0x0	Go against specification and start redistributors in woken-up state at reset. This allows software that was written for previous versions of the GICv3 specification to work correctly. This should not be used for production code or when the distributor is used separately from the core fast model.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

3.9.30 GICv3IRI_Filter

GICv3 metacomponent for redistribution of interrupts(contains Distributor, and configurable numbers of ITSs and Redistributors. Validation only version. This model is written in C++.

GICv3IRI_Filter contains the following CADI targets:

- GICv3IRI

GICv3IRI_Filter contains the following MTI components:

- *4.76 GICv3Distributor - trace* on page 4-2496
- *4.77 GICv3IRI - trace* on page 4-2504
- *4.80 GICv3Redistributor - trace* on page 4-2527
- *4.81 GICv3RedistributorInternal - trace* on page 4-2541

About GICv3IRI_Filter

The GICv3IRI_Filter has similar behavior to the GICv3IRI, except for the slave interface. Any transaction accessing a 4KB page that is not used by the GIC, as configurable through the parameters, is forwarded to the `pvbus_filtermiss_m` port, which is only present in this variant.

Note

- Set the `FASTSIM_GIC_MEMORY_MAP` environment variable to 1 to print to `stderr` the memory map of any GICv3 or later models that are included in the platform being run.
- Set the `GICD_ITARGETSR-RAZWI` parameter to true for legacy, GICv2-style routing, where interrupts are routed to the first processor in the system

Table 3-375 Ports

Name	Protocol	Type	Description
`cfgsdisable`	*Signal* on page 2-85	Slave	Disable some SPIs signal.
`extended_ppi_in_n[64]`	*Signal* on page 2-85	Slave	Extended private peripheral interrupts (ID1056-ID1119) for cpu *n*, where *n* is in the range 0-255.
`extended_spi_in[1024]`	*Signal* on page 2-85	Slave	Extended Shared peripheral interrupts.
`msi_error_interrupt`	*Signal* on page 2-85	Master	When report of MSI error allowed through interrupt, loop-back this signal to relevant IRQ input signal
`po_reset`	*Signal* on page 2-85	Slave	Resets.
`ppi_in_n[16]`	*Signal* on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu *n*, where *n* is in the range 0-255.
`pvbus_filtermiss_m`	`PVBus`	Master	Passthrough for accesses to pages not used by the GIC IRI.
`pvbus_m`	`PVBus`	Master	Memory bus for transactions generated by the GIC.
`pvbus_s`	`PVBus`	Slave	Memory bus in.
`redistributor_m[256]`	*GICv3Comms* on page 2-84	Master	Input from and output to CPU interface.
`reset`	*Signal* on page 2-85	Slave	Resets.
`spi_in[988]`	*Signal* on page 2-85	Slave	Shared peripheral interrupts.
`wake_request[256]`	*Signal* on page 2-85	Master	Power management outputs.
`wire_to_msi_0[1024]`	*Signal* on page 2-85	Slave	Wire-to-MSI interrupts for architectural consolidator 0.
`wire_to_msi_1[1024]`	*Signal* on page 2-85	Slave	Wire-to-MSI interrupts for architectural consolidator 1.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-375 Ports (continued)

Name	Protocol	Type	Description
wire_to_msi_2[1024]	*Signal* on page 2-85	Slave	Wire-to-MSI interrupts for architectural consolidator 2.
wire_to_msi_3[1024]	*Signal* on page 2-85	Slave	Wire-to-MSI interrupts for architectural consolidator 3.

Table 3-376 Parameters for GICv3IRI

Name	Type	Default value	Description
A3-affinity-supported	bool	0x0	Device supports affinity level 3 values that are non-zero.
ARE-fixed-to-one	bool	0x0	GICv2 compatibility is not supported and GICD_CTLR.ARE_* is always one
CPU-affinities	string	""	A comma separated list of dotted quads containing the affinities of all PEs connected to this IRI. If CPU-affinities-file is specified, this parameter is ignored.
CPU-affinities-file	string	""	A file containing comma separated list of dotted quads containing the affinities of all PEs connected to this IRI. If this parameter is specified, CPU-affinities parameter will be ignored even when it is given.
DPG-ARE-only	bool	0x0	Limit application of DPG bits to interrupt groups for which ARE=1
DPG-bits-implemented	bool	0x0	Enable implementation of interrupt group participation bits or DPG bits in GICR_CTLR
DS-fixed-to-zero	bool	0x0	Enable/disable support of single security state
GICD-alias	int	0x0	In GICv2 mode: the base address for a 4k page alias of the first 4k of the Distributor page, in GICv3 mode. the base address of a 64KB page containing message based SPI signalling register aliases(0:Disabled)
GICD-legacy-registers-as-reserved	bool	0x0	When ARE is RAO/WI, makes superfluous registers in GICD reserved (including for the purpose of STATUSR updates)
GICD_CTLR-DS-1-means-secure-only	bool	0x0	If GICD_CTLR.DS=1, GICD supports a single security state which is secure if this is true, otherwise is non-secure.
GICD_ITARGETSR-RAZWI	bool	0x0	If true, the GICD_ITARGETS registers are RAZ/WI
GICD_PIDR	int	0x0	The value for the GICD_PIDR registers, if non-zero. Note: fixed fields (device type etc.) will be overriden in this value.
GICR-clear-enable-supported	bool	0x0	When true, this sets the value of the RO bit GICR_CTLR.CES with the value of the parameter allow-LPIEN-clear, making it visible to software
GICR-invalidate-registers-implemented	bool	0x0	When true, the registers GICR_INVLPIR, GICR_INVALLR and GICR_SYNCR are implemented
GICR_PIDR	int	0x0	The value for the GICR_PIDR registers, if non-zero. Note: fixed fields (device type etc.) will be overriden in this value.
GICR_PROPBASER-read-only	bool	0x0	GICR_PROPBASER register is read-only.
GICR_PROPBASER-reset-value	int	0x0	Value of GICR_PROPBASER on reset.
GITS_BASER0-entry-bytes	int	0x8	Number of bytes required per entry for GITS_BASER0 register.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-376 Parameters for GICv3IRI (continued)

Name	Type	Default value	Description
GITS_BASER0-indirect-RAZ	bool	0x0	Indirect field for GITS_BASER0 register is RAZ/WI.
GITS_BASER0-type	int	0x0	Type field for GITS_BASER0 register. 0 = Unimplemented; 1 = Devices; 2 = Virtual Processors; 3 = Physical Processors; 4 = Collections
GITS_BASER1-entry-bytes	int	0x8	Number of bytes required per entry for GITS_BASER1 register.
GITS_BASER1-indirect-RAZ	bool	0x0	Indirect field for GITS_BASER1 register is RAZ/WI.
GITS_BASER1-type	int	0x0	Type field for GITS_BASER1 register. 0 = Unimplemented; 1 = Devices; 2 = Virtual Processors; 3 = Physical Processors; 4 = Collections
GITS_BASER2-entry-bytes	int	0x8	Number of bytes required per entry for GITS_BASER2 register.
GITS_BASER2-indirect-RAZ	bool	0x0	Indirect field for GITS_BASER2 register is RAZ/WI.
GITS_BASER2-type	int	0x0	Type field for GITS_BASER2 register. 0 = Unimplemented; 1 = Devices; 2 = Virtual Processors; 3 = Physical Processors; 4 = Collections
GITS_BASER3-entry-bytes	int	0x8	Number of bytes required per entry for GITS_BASER3 register.
GITS_BASER3-indirect-RAZ	bool	0x0	Indirect field for GITS_BASER3 register is RAZ/WI.
GITS_BASER3-type	int	0x0	Type field for GITS_BASER3 register. 0 = Unimplemented; 1 = Devices; 2 = Virtual Processors; 3 = Physical Processors; 4 = Collections
GITS_BASER4-entry-bytes	int	0x8	Number of bytes required per entry for GITS_BASER4 register.
GITS_BASER4-indirect-RAZ	bool	0x0	Indirect field for GITS_BASER4 register is RAZ/WI.
GITS_BASER4-type	int	0x0	Type field for GITS_BASER4 register. 0 = Unimplemented; 1 = Devices; 2 = Virtual Processors; 3 = Physical Processors; 4 = Collections
GITS_BASER5-entry-bytes	int	0x8	Number of bytes required per entry for GITS_BASER5 register.
GITS_BASER5-indirect-RAZ	bool	0x0	Indirect field for GITS_BASER5 register is RAZ/WI.
GITS_BASER5-type	int	0x0	Type field for GITS_BASER5 register. 0 = Unimplemented; 1 = Devices; 2 = Virtual Processors; 3 = Physical Processors; 4 = Collections
GITS_BASER6-entry-bytes	int	0x8	Number of bytes required per entry for GITS_BASER6 register.
GITS_BASER6-indirect-RAZ	bool	0x0	Indirect field for GITS_BASER6 register is RAZ/WI.
GITS_BASER6-type	int	0x0	Type field for GITS_BASER6 register. 0 = Unimplemented; 1 = Devices; 2 = Virtual Processors; 3 = Physical Processors; 4 = Collections
GITS_BASER7-entry-bytes	int	0x8	Number of bytes required per entry for GITS_BASER7 register.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-376 Parameters for GICv3IRI (continued)

Name	Type	Default value	Description
GITS_BASER7-indirect-RAZ	bool	0x0	Indirect field for GITS_BASER7 register is RAZ/WI.
GITS_BASER7-type	int	0x0	Type field for GITS_BASER7 register. 0 = Unimplemented; 1 = Devices; 2 = Virtual Processors; 3 = Physical Processors; 4 = Collections
GITS_PIDR	int	0x0	The value for the GITS_PIDR registers, if non-zero. Note: fixed fields (device type etc.) will be overriden in this value.
ICFGR-PPI-mask	int	0xaaaaaaaa	Mask for writes to ICFGR registers that configure PPIs
ICFGR-PPI-reset	int	0x0	Reset value for ICFGR regesters that configure PPIs
ICFGR-SGI-mask	int	0x0	Mask for writes to ICFGR registers that configure SGIs
ICFGR-SGI-reset	int	0xaaaaaaaa	Reset value for ICFGR registers that configure SGIs
ICFGR-SPI-mask	int	0xaaaaaaaa	Mask for writes to ICFGR registers that configure SPIs
ICFGR-SPI-reset	int	0x0	Reset value for ICFGR regesters that configure SPIs
ICFGR-rsvd-bit	bool	0x0	If ARE=0, the value of reserved bits i.e. bit 0,2,4..30 of ICFGRn for n>0
IGROUP-PPI-mask	int	0xffff	Mask for writes to PPI bits in IGROUP registers
IGROUP-PPI-reset	int	0x0	Reset value for SGI bits in IGROUP registers
IGROUP-SGI-mask	int	0xffff	Mask for writes to SGI bits in IGROUP registers
IGROUP-SGI-reset	int	0x0	Reset value for SGI bits in IGROUP registers
IIDR	int	0x0	GICD_IIDR and GICR_IIDR value
IRI-ID-bits	int	0x10	Number of bits used to represent interrupts IDs in the Distributor and Redistributors, forced to 10 when none of LPIs, extended SPIs or extended PPIs is supported
IROUTER-IRM-RAZ-WI	bool	0x0	GICD_IROUTERn.InterruptRoutingMode is RAZ/WI
ITS-BASER-force-page-alignement	bool	0x1	Force alignement of address writen to a GITS_BASER register to the page size configured
ITS-ID-bits	int	0x10	Number of interrupt bits supported by ITS.
ITS-MOVALL-update-collections	bool	0x0	Whether MOVALL command updates the collection entires
ITS-TRANSLATE64R	bool	0x0	Add an implementation specific register at 0x10008 supporting 64 bit TRANSLATER (dev[63:32], interupt[31:0])
ITS-collection-ID-bits	int	0x0	Number of collection bits supported by ITS (optional parameter, 0 => 16bits support and GITS_TYPER.CIL=0
ITS-count	int	0x0	Number of Interrupt Translation Services to be instantiated (0=none)
ITS-cumulative-collection-tables	bool	0x1	When true, the supported amount of collections is the sum of GITS_TYPER.HCC and the number of collections supported in memory, otherwise, simply the number supported in memory only. Irrelevant when HCC=0
ITS-device-bits	int	0x10	Number of bits supported for ITS device IDs.
ITS-entry-size	int	0x8	Number of bytes required to store each entry in the ITT tables.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-376 Parameters for GICv3IRI (continued)

Name	Type	Default value	Description
ITS-hardware-collection-count	int	0x0	Number of hardware collections held exclusively in the ITS
ITS-legacy-iidr-typer-offset	bool	0x0	Put the GITS_IIDR and GITS_TYPER registers at their older offset of 0x8 and 0x4 respectively
ITS-shared-vPE-table	int	0x0	Number of affinity levels to which the vPE configuration table is shared. This parameter is valid when has-gicv4.1 is true.
ITS-threaded-command-queue	bool	0x1	Enable execution of ITS commands in a separate thread which is sometimes required for cosimulation
ITS-use-physical-target-addresses	bool	0x1	Use physical hardware adresses for targets in ITS commands -- must be true for distributed implementations
ITS-vmovp-bit	bool	0x0	Device supports software issuing a VMOVP to only one of the ITSs that has a mapping for a vPE. The device itself ensures synchronization of the VMOVP command across all ITSs that have mapping for that vPE.
ITS0-base	int	0x0	Register base address for ITS0 (automatic if 0).
ITS1-base	int	0x0	Register base address for ITS1 (automatic if 0).
ITS2-base	int	0x0	Register base address for ITS2 (automatic if 0).
ITS3-base	int	0x0	Register base address for ITS3 (automatic if 0).
LPI-cache-check-data	bool	0x0	Enable Cached LPI data against memory checking when available for cache type
LPI-cache-type	int	0x1	Cache type for LPIs, 0:No caching, 1:Full caching
MSI_IIDR	int	0x0	Value returned in MSI_IIDR registers.
MSI_NS-frame0-base	int	0x0	If non-zero, sets the base address used for non-secure MSI frame 0 registers.
MSI_NS-frame0-max-SPI	int	0x0	Maximum SPI ID supported by non-secure MSI frame 0. Set to 0 to disable frame.
MSI_NS-frame0-min-SPI	int	0x0	Minimum SPI ID supported by non-secure MSI frame 0. Set to 0 to disable frame.
MSI_NS-frame1-base	int	0x0	If non-zero, sets the base address used for non-secure MSI frame 1 registers.
MSI_NS-frame1-max-SPI	int	0x0	Maximum SPI ID supported by non-secure MSI frame 1. Set to 0 to disable frame.
MSI_NS-frame1-min-SPI	int	0x0	Minimum SPI ID supported by non-secure MSI frame 1. Set to 0 to disable frame.
MSI_NS-frame2-base	int	0x0	If non-zero, sets the base address used for non-secure MSI frame 2 registers.
MSI_NS-frame2-max-SPI	int	0x0	Maximum SPI ID supported by non-secure MSI frame 2. Set to 0 to disable frame.
MSI_NS-frame2-min-SPI	int	0x0	Minimum SPI ID supported by non-secure MSI frame 2. Set to 0 to disable frame.
MSI_NS-frame3-base	int	0x0	If non-zero, sets the base address used for non-secure MSI frame 3 registers.
MSI_NS-frame3-max-SPI	int	0x0	Maximum SPI ID supported by non-secure MSI frame 3. Set to 0 to disable frame.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-376 Parameters for GICv3IRI (continued)

Name	Type	Default value	Description
MSI_NS-frame3-min-SPI	int	0x0	Minimum SPI ID supported by non-secure MSI frame 3. Set to 0 to disable frame.
MSI_NS-frame4-base	int	0x0	If non-zero, sets the base address used for non-secure MSI frame 4 registers.
MSI_NS-frame4-max-SPI	int	0x0	Maximum SPI ID supported by non-secure MSI frame 4. Set to 0 to disable frame.
MSI_NS-frame4-min-SPI	int	0x0	Minimum SPI ID supported by non-secure MSI frame 4. Set to 0 to disable frame.
MSI_NS-frame5-base	int	0x0	If non-zero, sets the base address used for non-secure MSI frame 5 registers.
MSI_NS-frame5-max-SPI	int	0x0	Maximum SPI ID supported by non-secure MSI frame 5. Set to 0 to disable frame.
MSI_NS-frame5-min-SPI	int	0x0	Minimum SPI ID supported by non-secure MSI frame 5. Set to 0 to disable frame.
MSI_NS-frame6-base	int	0x0	If non-zero, sets the base address used for non-secure MSI frame 6 registers.
MSI_NS-frame6-max-SPI	int	0x0	Maximum SPI ID supported by non-secure MSI frame 6. Set to 0 to disable frame.
MSI_NS-frame6-min-SPI	int	0x0	Minimum SPI ID supported by non-secure MSI frame 6. Set to 0 to disable frame.
MSI_NS-frame7-base	int	0x0	If non-zero, sets the base address used for non-secure MSI frame 7 registers.
MSI_NS-frame7-max-SPI	int	0x0	Maximum SPI ID supported by non-secure MSI frame 7. Set to 0 to disable frame.
MSI_NS-frame7-min-SPI	int	0x0	Minimum SPI ID supported by non-secure MSI frame 7. Set to 0 to disable frame.
MSI_PIDR	int	0x0	The value for the MSI_PIDR registers, if non-zero and distributor supports GICv2m. Note: fixed fields (device type etc.) will be overriden in this value.
MSI_S-frame0-base	int	0x0	If non-zero, sets the base address used for secure MSI frame 0 registers.
MSI_S-frame0-max-SPI	int	0x0	Maximum SPI ID supported by secure MSI frame 0. Set to 0 to disable frame.
MSI_S-frame0-min-SPI	int	0x0	Minimum SPI ID supported by secure MSI frame 0. Set to 0 to disable frame.
MSI_S-frame1-base	int	0x0	If non-zero, sets the base address used for secure MSI frame 1 registers.
MSI_S-frame1-max-SPI	int	0x0	Maximum SPI ID supported by secure MSI frame 1. Set to 0 to disable frame.
MSI_S-frame1-min-SPI	int	0x0	Minimum SPI ID supported by secure MSI frame 1. Set to 0 to disable frame.
MSI_S-frame2-base	int	0x0	If non-zero, sets the base address used for secure MSI frame 2 registers.
MSI_S-frame2-max-SPI	int	0x0	Maximum SPI ID supported by secure MSI frame 2. Set to 0 to disable frame.
MSI_S-frame2-min-SPI	int	0x0	Minimum SPI ID supported by secure MSI frame 2. Set to 0 to disable frame.
MSI_S-frame3-base	int	0x0	If non-zero, sets the base address used for secure MSI frame 3 registers.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-376 Parameters for GICv3IRI (continued)

Name	Type	Default value	Description
MSI_S-frame3-max-SPI	int	0x0	Maximum SPI ID supported by secure MSI frame 3. Set to 0 to disable frame.
MSI_S-frame3-min-SPI	int	0x0	Minimum SPI ID supported by secure MSI frame 3. Set to 0 to disable frame.
MSI_S-frame4-base	int	0x0	If non-zero, sets the base address used for secure MSI frame 4 registers.
MSI_S-frame4-max-SPI	int	0x0	Maximum SPI ID supported by secure MSI frame 4. Set to 0 to disable frame.
MSI_S-frame4-min-SPI	int	0x0	Minimum SPI ID supported by secure MSI frame 4. Set to 0 to disable frame.
MSI_S-frame5-base	int	0x0	If non-zero, sets the base address used for secure MSI frame 5 registers.
MSI_S-frame5-max-SPI	int	0x0	Maximum SPI ID supported by secure MSI frame 5. Set to 0 to disable frame.
MSI_S-frame5-min-SPI	int	0x0	Minimum SPI ID supported by secure MSI frame 5. Set to 0 to disable frame.
MSI_S-frame6-base	int	0x0	If non-zero, sets the base address used for secure MSI frame 6 registers.
MSI_S-frame6-max-SPI	int	0x0	Maximum SPI ID supported by secure MSI frame 6. Set to 0 to disable frame.
MSI_S-frame6-min-SPI	int	0x0	Minimum SPI ID supported by secure MSI frame 6. Set to 0 to disable frame.
MSI_S-frame7-base	int	0x0	If non-zero, sets the base address used for secure MSI frame 7 registers.
MSI_S-frame7-max-SPI	int	0x0	Maximum SPI ID supported by secure MSI frame 7. Set to 0 to disable frame.
MSI_S-frame7-min-SPI	int	0x0	Minimum SPI ID supported by secure MSI frame 7. Set to 0 to disable frame.
PA_SIZE	int	0x30	Number of valid bits in physical address
PPI-implemented-mask	int	0xffff	Mask of PPIs that are implemented. One bit per PPI bit 0 == PPI 16 (first PPI). This will affect other masks.
SPI-count	int	0xe0	Number of SPIs that are implemented.
SPI-message-based-support	bool	0x1	Distributor supports meassage based signaling of SPI
SPI-unimplemented	string	""	A comma spearated list of unimplemented SPIs ranges for sparse SPI defintion(for ex: '35, 39-42, 73)'
STATUSR-implemented	bool	0x1	Determines whether the GICR_STATUSR register is implemented.
allow-LPIEN-clear	bool	0x0	Allow RW behaviour on GICR_CTLR.LPIEN isntead of set once
chip-id	int	0x0	Chip ID when multichip operation is enabled.
common-lpi-configuration	int	0x0	Describes which re-distributors share (and must be configured with the same) LPI configuration table as described in GICR_TYPER(0:All, 1:A.x.x.x, 2:A.B.x.x, 3:A.B.C.x

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-376 Parameters for GICv3IRI (continued)

Name	Type	Default value	Description
common-vPE-table-affinity	string	""	Affinity value list in the form of 'a.b.c.d, e.f.g.h, etc' given to the ITS(s) (where a.b.c.d corresponds to ITS0, e.f.g.h corresponds to ITS1 and so on). Under an affinity value the vPE configuration table is shared among redistributors where the level to be shared is defined by ITS-shared-vPE-table. This parameter is valid when has-gicv4.1 is true.
consolidators	string	""	Specify consolidators' base addresses, interrupt line counts and base interrupt IDs, in the form 'baseAddr0:itlineCount0:baseINTID0, baseAddr1:itlineCount1:baseINTID1, [etc]' (eg '0x3f100000:64:4096, 0x3f200000:64:4224'). The consolidators' count is inferred from the list (maximum of 4). If not specified, the component contains no consolidators.
delay-ITS-accesses	bool	0x1	Delay accesses from the ITS until GICR_SYNCR is read.
delay-redistributor-accesses	bool	0x1	Delay memory accesses from the redistributor until GICR_SYNCR is read.
direct-lpi-support	bool	0x0	Enable support for LPI operations through GICR registers
enable-multichip-operation	bool	0x0	Enables multi-chip operation between Distributors in distributed GIC IRI
enable_protocol_checking	bool	0x0	Enable/disable protocol checking at cpu interface
enabled	bool	0x1	Enable GICv3 functionality; when false the component is inactive.
extended-ppi-count	int	0x0	Number of extended PPI supported
extended-spi-count	int	0x0	Number of extended SPI supported
fixed-routed-spis	string	""	Value of IROUTER[n] register in the form 'n=a.b.c.d, n=*'. The RM bit of IROUTER is 0 when n=a.b.c.d is used else 1 when n=* is used. n can be >= 32 and <= 1019
gicv2-only	bool	0x0	If true, when using the GICv3 model, pretend to be a GICv2 system
has-two-security-states	bool	0x1	If true, has two security states
has_VPENDBASER-dirty-flag-on-load	bool	0x0	GICR_VPENDBASER.Dirty reflects transient loading state when valid=1
has_mpam	bool	0x0	Enable MPAM support on ITS and RDs
ignore-generate-sgi-when-no-are	bool	0x0	Ignore GenerateSGI packets coming form the CPU interface if both ARE_S and ARE_NS are 0
irouter-default-mask	string	""	Default Mask value for IROUTER[32..1019] register in the form 'a.b.c.d'
irouter-default-reset	string	""	Default Reset Value of IROUTER[32..1019] register in the form 'a.b.c.d' or *
irouter-mask-values	string	""	Mask Value of IROUTER[n] register in the form 'n=a.b.c.d'.n can be >= 32 and <= 1019
irouter-reset-values	string	""	Reset Value of IROUTER[n] register in the form 'n=a.b.c.d or n=*'.n can be >= 32 and <= 1019
legacy-sgi-enable-rao	bool	0x0	Enables for SGI associated with an ARE=0 regime are RAO/WI
local-SEIs	bool	0x0	Generate SEI to signal internal issues

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-376 Parameters for GICv3IRI (continued)

Name	Type	Default value	Description
local-VSEIs	bool	0x0	Generate VSEI to signal internal issues
lockable-SPI-count	int	0x0	Number of SPIs that are locked down when CFGSDISABLE signal is asserted. Only applies for GICv2.
max-pe-on-chip	int	0x4	Maximum number of cores on any single chip. This will be used to identify the target chip and core when multichip operation is enabled.
monolithic	bool	0x0	Indicate that the implementation is not distributed
mpam_max_partid	int	0xffff	Maximum valid PARTID
mpam_max_pmg	int	0xff	Maximum valid PMG
multichip-threaded-dgi	bool	0x1	Enable sending multichip DGI messages in a separate thread. This parameter is valid only if enable-multichip-operation is set.
non-ARE-core-count	int	0x8	Maximum number of non-ARE cores; normally used to pass the cluster-level NUM_CORES parameter to the top-level redistributor.
outer-cacheability-support	bool	0x0	Allow configuration of outer cachability attributes in ITS and Redistributor
output_attributes	string	"ExtendedID[62:55]=MPAM_PMG, ExtendedID[54:39]=MPAM_PARTID, ExtendedID[38]=MPAM_NS"	User-defined transform to be applied to bus attributes like MasterID, ExtendedID or UserFlags. Currently, only works for MPAM Attributes encoding into bus attributes.
print-memory-map	bool	0x0	Print memory map to stdout
priority-bits	int	0x5	Number of implemented priority bits
processor-numbers	string	""	Specify processor numbers (as appears in GICR_TYPER) in the form 0.0.0.0=0,0.0.0.1=1 etc.) If not specified, will number processors starting at 0.
redistributor-threaded-sync	bool	0x1	Enable execution of redistributor delayed transactions in a separate thread which is sometimes required for cosimulation
reg-base	int	0x2c010000	Base for decoding GICv3 registers.
reg-base-per-redistributor	string	""	Base address for each redistributor in the form '0.0.0.0=0x2c010000, 0.0.0.1=0x2c020000'. All redistributors must be specified and this overrides the reg-base parameter (except that reg-base will still be used for the top-level redistributor). If reg-base-per-redistributor-file is specified, this parameter is ignored.
reg-base-per-redistributor-file	string	""	Path to file containing the base address for each redistributor in the form '0.0.0.0=0x2c010000, 0.0.0.1=0x2c020000'. All redistributors must be specified and this overrides the reg-base parameter (except that reg-base will still be used for the top-level redistributor). If this parameter is specified, reg-base-per-redistributor parameter will be ignored even when it is given.
report-MSI-error-via-statusr	int	0x0	Report MSI error via GITS_STATUSR. (0:unsupported, 1:report by GITS_STATUSR, 2:report by GITS_STATUSR and interrupt as well).

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-376 Parameters for GICv3IRI (continued)

Name	Type	Default value	Description
`sgi-range-selector-support`	`bool`	`0x0`	Device has support for the Range Selector feature for SGI
`single-set-support`	`bool`	`0x0`	When true, forces redistributors to recall interrupts with a clear rather than issue a second Set command
`supports-shareability`	`bool`	`0x1`	Device supports shareability attributes on outgoing memory bus (i.e. is modelling an ACElite port rather than an AXI4 port).
`trace-speculative-lpi-property-update`	`bool`	`0x0`	Trace LPI propery updates performed on speculative accesses (useful for debuging LPI)
`virtual-lpi-support`	`bool`	`0x0`	GICv4 Virtual LPIs and Direct injection of Virtual LPIs supported
`virtual-priority-bits`	`int`	`0x5`	Number of implemented virtual priority bits
`wakeup-on-reset`	`bool`	`0x0`	Go against specification and start redistributors in woken-up state at reset. This allows software that was written for previous versions of the GICv3 specification to work correctly. This should not be used for production code or when the distributor is used separately from the core fast model.

3.9.31 ICS307

Serially Programmable Clock Source. This model is written in LISA+.

ICS307 contains the following CADI targets:

- ClockDivider
- ICS307

ICS307 contains the following MTI components:

- *4.60 ClockDivider - trace* on page 4-2438

About ICS307

You can use this component to convert the rate of one ClockSignal to another ClockSignal by using configurable multiplier, divider, and scale values. The divider ratio can be set by startup parameters or at runtime by a configuration port. Changes to the input ClockSignal rate and divider ratio are reflected immediately by the output ClockSignal ports.

Three values determine the divisor ratio:

- `vdw`.
- `rdw`.
- `od`.

To calculate the divisor ratio, use:

```
Divisor = ((rdw+2) * scale) / (2 * (vdw+8))
```

where `scale` is derived from this table indexed by `od`:

Table 3-377 od to scale conversion

od	scale
0	10
1	2
2	8

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-377 od to scale conversion (continued)

od	scale
3	4
4	5
5	7
6	3
7	6

The default values of `vdw`, `rdw` and `od` are 4, 6 and 3 to give a default divisor rate of:

```
((6+2) * 4) / (2 * (4+8)) = 4/3
```

Table 3-378 Ports

Name	Protocol	Type	Description
`clk_in`	`ClockSignal`	Slave	Master clock rate.
`clk_out_clk1`	`ClockSignal`	Master	Modified clock rate.
`clk_out_ref`	`ClockSignal`	Master	Pass through of master clock rate for divider chaining.
`configuration`	*ICS307Configuration on page 2-72*	Slave	Configuration port for setting divider ratio dynamically.

Table 3-379 Parameters for ICS307

Name	Type	Default value	Description
`od`	`int`	`0x3`	OD
`rdr`	`int`	`0x6`	RDR
`vdw`	`int`	`0x4`	VDW

3.9.32 IDAU

IDAU is device which provides Security attribute relating to the address pass to it. For each memory access (data and instruction), the CPU checks the IDAU and sets the security of its transactions based on it. This model is written in C++.

About IDAU

An Implementation Defined Attribution Unit (*IDAU*) is a device that provides a security attribute relating to an address passed to it. For each memory access (data and instructions), the CPU checks the IDAU and sets the security of its transactions based on it.

The IDAU model uses the `pv::IDAUSignal` struct to return the security attributes for the address that is passed to it. Unlike the hardware, the CPU Fast Model does not query the IDAU for each access. Communication is at a higher abstraction level to maintain simulation speed.

The fields of the `pv::IDAUSignal` struct map to the hardware signals as follows:

Table 3-380 Mappings between pv::IDAUSignal fields and hardware signals

Hardware signal	`pv::IDAUSignal` field	Description
`IDAUNS`	`bool ns`	Non-secure region response.
`IDAUNSC`	`bool nsc`	Non-secure-callable region response.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-380 Mappings between pv::IDAUSignal fields and hardware signals (continued)

Hardware signal	pv::IDAUSignal field	Description
IDAUID	uint8_t region	Region number.
IDAUIDV	bool valid	Region number valid.
IDAUNCHK	bool exempt	Region exempt from attribution check.

Masters can read or write to the pvbus_s port as follows:

Read

Reads return the IDAU region's pv::IdauRegion struct (32 bytes), which contains information about the IDAU region for the requested address. pv::IdauRegion contains the start address, end address, pv::IDAUSignal and 8 bytes of padding to make it 32-byte aligned.

Write

The port only supports 32-byte write operations to pass in a pv::IdauRegion struct for updating an internal IDAU region.

DMI

The port adds support for DMI requests and provides a pointer to a pv::IdauRegion for the requested address. An invalid DMI call back occurs if the IDAU updates its regions.

Note

To disable the IDAU, set the NUM_IDAU_REGION parameter to zero.

Table 3-381 Ports

Name	Protocol	Type	Description
invalidate_region	*Value_64 on page 2-85*	Master	This port is used as a call back to inform masters that the IDAU has updated its region information.
pvbus_s	PVBus	Slave	-

3.9.33 MMC

Generic Multimedia Card. This model is written in LISA+.

MMC contains the following CADI targets:

- MMC

About MMC

This component simulates an SD or SDHC card that is compatible with the *MultiMedia Card Association* (MMCA, *https://www.jedec.org*) specification version 3.31. The parameters permit configuration of a number of attributes reflected in the CID and CSD registers. You can customize the component further by modifying the supplied MMC model source code directly.

When paired with a PL180_MCI component, the MMC device model provides emulation of a flexible, persistent storage mechanism.

The MMC component uses a file on the host PC to simulate the storage device. The size of this backing store file determines the reported size of the MMC device. As small sections of this file are paged in by the model, large filesystems can be modeled while making efficient use of host PC memory. The backing store file can contain a partition table and filesystems such as FAT or EXT2.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

The image file is a direct bit copy of the contents of an SD card. If the image file that the `p_mmc_file` parameter refers to does not exist, the component behaves as if the card is absent. If the image file is read-only, then the component behaves as if the card is read-only.

Note

Operating system boots often attempt to write to the boot filesystem. They might not work properly if the boot filesystem is on a read-only card.

The MMC component does not model card insertion or removal. It models the card having already been inserted at system instantiation time.

You can configure the MMC component to behave as an SDHC card by setting the `card_type` parameter to `SDHC`. SDHC mode is a model-specific extension, and is not supported by PL180 hardware. It supports filesystems that are larger than 2GB.

The component supports these commands:

- `MMC_GO_IDLE_STATE`.
- `MMC_SEND_OP_COND`.
- `MMC_ALL_SEND_CID`.
- `MMC_SET_RELATIVE_ADDR`.
- `MMC_SET_DSR`.
- `MMC_SELDESL_CARD`.
- `MMC_SEND_CSD`.
- `MMC_SEND_CID`.
- `MMC_STOP_TRANSMISSION`.
- `MMC_SEND_STATUS`.
- `MMC_GO_INACTIVE_STATE`.
- `MMC_READ_SINGLE_BLOCK`.
- `MMC_READ_MULTIPLE_BLOCK`.
- `MMC_SET_BLOCK_COUNT`.
- `MMC_WRITE_BLOCK`.
- `MMC_WRITE_MULTIPLE_BLOCK`.
- `MMC_SEND_EXT_CSD`. This command is supported in SDHC mode only.

The block length is 512 bytes. SimGen reports attempts to change it as errors.

The component supports these erase commands (Class 5), but they have no effect on the disk backing storage:

- `MMC_ERASE_GROUP_START`.
- `MMC_ERASE_GROUP_END`.
- `MMC_ERASE`.

The component does not support these commands:

- `MMC_BUSTEST_R`.
- `MMC_BUSTEST_W`.

The component does not support stream read and write commands (Classes 1 and 3):

- `MMC_READ_DAT_UNTIL_STOP`.
- `MMC_WRITE_DAT_UNTIL_STOP`.
- `MMC_PROGRAM_CID`.
- `MMC_PROGRAM_CSD`.

The component does not support block oriented write protection commands (Class 6):

- `MMC_SET_WRITE_PROT`.
- `MMC_CLR_WRITE_PROT`.
- `MMC_SEND_WRITE_PROT`.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

The component does not support lock card commands (Class 7) or application-specific commands (Class 8):

- `MMC_LOCK_UNLOCK`.
- `MMC_APP_CMD`.
- `MMC_GEN_CMD`.

The component does not support I/O mode commands (Class 9):

- `MMC_FAST_IO`.
- `MMC_GO_IRQ_STATE`.

The component does not support reserved commands. Using a reserved command sets the MMC `ST_ER_B_ILLEGAL_COMMAND` bit in the status register of the card. Read this with the `MMC_SEND_STATUS` command.

Use the `p_diagnostics` parameter to select the level of diagnostic output, to help to debug device driver and controller-to-card protocol issues. It supports the following levels:

Level 0
None.

Level 1
Warnings about attempting to change read-only settings.

Level 2
Trace of command calls.

Level 3
Information about every step in the MMC_Protocol interaction.

Level 4
Hex dump of every block sent or received.

The registers are not memory mapped. Instead, you access them using relevant MMC commands. The MMC component model makes the registers available through a CADI interface. Modification of these registers through CADI is not recommended, but not prohibited. For example, modifying the card ID (CID) registers can be useful when experimenting with drivers, but direct modification of the STATUS_REG register is likely to put the card model into an indeterminate state.

For a full definition of MMC registers, see the MMCA System Summary documentation. Device-specific register information can also be obtained from MMC vendors.

Table 3-382 MMC registers

Name	CADI register number	Description
OCR_REG	`0x000`	Operating conditions register
CID_REG0	`0x004`	Card ID bits 127:96
CID_REG1	`0x005`	Card ID bits 95:64
CID_REG2	`0x006`	Card ID bits 63:32
CID_REG3	`0x007`	Card ID bits 31:0
CSD_REG0	`0x008`	Card specific data bits 127:96
CSD_REG1	`0x009`	Card specific data bits 95:64
CSD_REG2	`0x00a`	Card specific data bits 63:32
CSD_REG3	`0x00b`	Card specific data bit 31:0
RCA_REG	`0x00c`	Relative card address register
DSR_REG	`0x00d`	Driver stage register

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-382 MMC registers (continued)

Name	CADI register number	Description
BLOCKLEN_REG	0x00e	Block length
STATUS_REG	0x00f	Card status
BLOCK_COUNT_REG	0x010	Block count

Table 3-383 Ports

Name	Protocol	Type	Description
card_present	*StateSignal* on page 2-85	Master	Used to signal whether an MMC image is loaded. It is set if an image is loaded, and is clear if no image is loaded.
clk_in	ClockSignal	Slave	Input clock signal used to drive our 'bus'.
mmc	*MMC_Protocol* on page 2-73	Slave	The MMC slave port.

Table 3-384 Parameters for MMC

Name	Type	Default value	Description
card_type	string	"SD"	Card type ('SD' or 'SDHC')
diagnostics	int	0x0	Diagnostics level
force_sector_addressing	bool	0x0	Use sector addressing even on small cards
p_OEMid	int	0x0	Card ID OEM ID
p_fast_access	bool	0x1	Don't simulate MMC block access delays
p_manid	int	0x2	Card ID Manufacturer ID
p_mmc_file	string	"mmc.dat"	MMCard filename
p_prodName	string	"ARMmmc"	Card ID Product Name (6 chars)
p_prodRev	int	0x1	Card ID Product Revision
p_sernum	int	0xca4d0001	Card Serial Number

3.9.34 MMU_400

MMU-400 component. This model is written in LISA+.

MMU_400 contains the following CADI targets:

- MMU_400
- MMU_400_BASE

MMU_400 contains the following MTI components:

- *4.82 MMU_400_BASE - trace* on page 4-2547
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.106 PVBusMaster - trace* on page 4-2759
- *4.107 PVBusSlave - trace* on page 4-2760

About MMU_400

Set the `use_label_mapping` parameter to `true` if your upstream devices have labels in the top 16 bits of the transaction MasterID.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Note

The model does not have a concept of AXI-ID, but a transaction can have a MasterID set on it.

Label your upstream components 0…*N* so that the parameters of this component can map those integers to StreamID and SSD_Index.

Set `use_label_mapping` to `false` if the StreamID is encoded in the top 16 bits of the MasterID and the bottom 16 bits encode either the SSD_Index or the SSD state directly, depending on `use_ssd_determination_table`. Typically in hardware, a device emits different AXI-IDs, depending on what it is doing. In the model, MasterIDs are usually not diverse and a device might only emit one MasterID.

If `use_ssd_determination_table` is `true`, the bottom 16 bits of the MasterID encode the SSD_Index. They must be < 2^`ssd_index_width`. If it is `false`, they encode the SSD state directly (zero is Secure and nonzero is Non-secure).

This component models all architectural registers that are specified in the Technical Reference Manual (TRM), except that it does not model any of the performance registers, and has the following limitations:

- MMU-400 does not have an SMMU_STLBGSTATUS register because the Secure side is a nominal pass-through. MMU-400 only has stage 2 support and you cannot use stage 2 on the Secure side.
- The SMMU_NSACR is an alias of the Non-secure SMMU_ACR. This component models SMMU_ACR as RAZ/WI.
- The *ACR registers have IMP DEF contents. This component models only the PAGESIZE bit of the SACR, as non-RAZ/WI. It models no other IMP DEF registers.

Table 3-385 Ports

Name	Protocol	Type	Description
`apb3_control_ns`	`PVBus`	Slave	APBv3 control port for Non-secure access to the register file. If this port is used do not use the APBv4 port.
`apb3_control_s`	`PVBus`	Slave	APBv3 control port for Secure access to the register file. If this port is used do not use the APBv4 port.
`apb4_control`	`PVBus`	Slave	APBv4 control port for access to the register file. If this port is used do not use the APBv3 ports.
`cfg_cttw_in`	*Signal* on page 2-85	Slave	Enables coherent page table walks.
`cfg_flt_irpt_ns`	*Signal* on page 2-85	Master	Non-secure configuration access fault interrupt. Corresponds to SMMU architectural signal SMMU_NSgCfgIrpt.
`cfg_flt_irpt_s`	*Signal* on page 2-85	Master	Secure configuration access fault interrupt. Corresponds to SMMU architectural signal SMMU_gCfgIrpt.
`comb_irpt_ns`	*Signal* on page 2-85	Master	Non-secure combined interrupt.
`comb_irpt_s`	*Signal* on page 2-85	Master	Secure combined interrupt.
`cxt_irpt_ns`	*Signal* on page 2-85	Master	Non-secure context bank fault.
`glbl_flt_irpt_ns`	*Signal* on page 2-85	Master	Global Non-secure fault interrupt. Corresponds to SMMU architectural signal SMMU_NSgIrpt.
`glbl_flt_irpt_s`	*Signal* on page 2-85	Master	Global Secure fault interrupt. Corresponds to SMMU architectural signal SMMU_gIrpt.
`priv_internals`	`MMU_400_Internals`	Slave	For internal use only, please do not use.
`pvbus_m`	`PVBus`	Master	Downstream port of the MMU, where translated transactions emerge.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-385 Ports (continued)

Name	Protocol	Type	Description
pvbus_ptw_m	PVBus	Master	Downstream port for page table walks if configured using the ptw_has_separate_port parameter.
pvbus_s	PVBus	Slave	Upstream port of the MMU. Addresses on the port are in VA/IPA.
reset_in	*Signal* on page 2-85	Slave	Signal to reset the MMU.

Table 3-386 Parameters for MMU_400

Name	Type	Default value	Description
always_secure_ssd_indices	string	""	Non-programmable SSD Indexes that are always secure (e.g. 0, 6, 35-64).
cfg_cttw	bool	0x1	Perform coherent page table walks
dump_unpredictablity_in_user_flags	bool	0x0	Override the user flags to encode unpredictable information (validation only)
label0_read_ssd	int	0x0	Label0: Read SDD or SSD_Index
label0_read_stream_id	int	0x0	Label0: Read Stream ID
label0_write_ssd	int	0x0	Label0: Write SDD or SSD_Index
label0_write_stream_id	int	0x0	Label0: Write Stream ID
label10_read_ssd	int	0x0	Label10: Read SDD or SSD_Index
label10_read_stream_id	int	0x0	Label10: Read Stream ID
label10_write_ssd	int	0x0	Label10: Write SDD or SSD_Index
label10_write_stream_id	int	0x0	Label10: Write Stream ID
label11_read_ssd	int	0x0	Label11: Read SDD or SSD_Index
label11_read_stream_id	int	0x0	Label11: Read Stream ID
label11_write_ssd	int	0x0	Label11: Write SDD or SSD_Index
label11_write_stream_id	int	0x0	Label11: Write Stream ID
label12_read_ssd	int	0x0	Label12: Read SDD or SSD_Index
label12_read_stream_id	int	0x0	Label12: Read Stream ID
label12_write_ssd	int	0x0	Label12: Write SDD or SSD_Index
label12_write_stream_id	int	0x0	Label12: Write Stream ID
label13_read_ssd	int	0x0	Label13: Read SDD or SSD_Index
label13_read_stream_id	int	0x0	Label13: Read Stream ID
label13_write_ssd	int	0x0	Label13: Write SDD or SSD_Index
label13_write_stream_id	int	0x0	Label13: Write Stream ID
label14_read_ssd	int	0x0	Label14: Read SDD or SSD_Index
label14_read_stream_id	int	0x0	Label14: Read Stream ID
label14_write_ssd	int	0x0	Label14: Write SDD or SSD_Index
label14_write_stream_id	int	0x0	Label14: Write Stream ID

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-386 Parameters for MMU_400 (continued)

Name	Type	Default value	Description
label15_read_ssd	int	0x0	Label15: Read SDD or SSD_Index
label15_read_stream_id	int	0x0	Label15: Read Stream ID
label15_write_ssd	int	0x0	Label15: Write SDD or SSD_Index
label15_write_stream_id	int	0x0	Label15: Write Stream ID
label16_read_ssd	int	0x0	Label16: Read SDD or SSD_Index
label16_read_stream_id	int	0x0	Label16: Read Stream ID
label16_write_ssd	int	0x0	Label16: Write SDD or SSD_Index
label16_write_stream_id	int	0x0	Label16: Write Stream ID
label17_read_ssd	int	0x0	Label17: Read SDD or SSD_Index
label17_read_stream_id	int	0x0	Label17: Read Stream ID
label17_write_ssd	int	0x0	Label17: Write SDD or SSD_Index
label17_write_stream_id	int	0x0	Label17: Write Stream ID
label18_read_ssd	int	0x0	Label18: Read SDD or SSD_Index
label18_read_stream_id	int	0x0	Label18: Read Stream ID
label18_write_ssd	int	0x0	Label18: Write SDD or SSD_Index
label18_write_stream_id	int	0x0	Label18: Write Stream ID
label19_read_ssd	int	0x0	Label19: Read SDD or SSD_Index
label19_read_stream_id	int	0x0	Label19: Read Stream ID
label19_write_ssd	int	0x0	Label19: Write SDD or SSD_Index
label19_write_stream_id	int	0x0	Label19: Write Stream ID
label1_read_ssd	int	0x0	Label1: Read SDD or SSD_Index
label1_read_stream_id	int	0x0	Label1: Read Stream ID
label1_write_ssd	int	0x0	Label1: Write SDD or SSD_Index
label1_write_stream_id	int	0x0	Label1: Write Stream ID
label20_read_ssd	int	0x0	Label20: Read SDD or SSD_Index
label20_read_stream_id	int	0x0	Label20: Read Stream ID
label20_write_ssd	int	0x0	Label20: Write SDD or SSD_Index
label20_write_stream_id	int	0x0	Label20: Write Stream ID
label21_read_ssd	int	0x0	Label21: Read SDD or SSD_Index
label21_read_stream_id	int	0x0	Label21: Read Stream ID
label21_write_ssd	int	0x0	Label21: Write SDD or SSD_Index
label21_write_stream_id	int	0x0	Label21: Write Stream ID
label22_read_ssd	int	0x0	Label22: Read SDD or SSD_Index
label22_read_stream_id	int	0x0	Label22: Read Stream ID
label22_write_ssd	int	0x0	Label22: Write SDD or SSD_Index

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-386 Parameters for MMU_400 (continued)

Name	Type	Default value	Description
label22_write_stream_id	int	0x0	Label22: Write Stream ID
label23_read_ssd	int	0x0	Label23: Read SDD or SSD_Index
label23_read_stream_id	int	0x0	Label23: Read Stream ID
label23_write_ssd	int	0x0	Label23: Write SDD or SSD_Index
label23_write_stream_id	int	0x0	Label23: Write Stream ID
label24_read_ssd	int	0x0	Label24: Read SDD or SSD_Index
label24_read_stream_id	int	0x0	Label24: Read Stream ID
label24_write_ssd	int	0x0	Label24: Write SDD or SSD_Index
label24_write_stream_id	int	0x0	Label24: Write Stream ID
label25_read_ssd	int	0x0	Label25: Read SDD or SSD_Index
label25_read_stream_id	int	0x0	Label25: Read Stream ID
label25_write_ssd	int	0x0	Label25: Write SDD or SSD_Index
label25_write_stream_id	int	0x0	Label25: Write Stream ID
label26_read_ssd	int	0x0	Label26: Read SDD or SSD_Index
label26_read_stream_id	int	0x0	Label26: Read Stream ID
label26_write_ssd	int	0x0	Label26: Write SDD or SSD_Index
label26_write_stream_id	int	0x0	Label26: Write Stream ID
label27_read_ssd	int	0x0	Label27: Read SDD or SSD_Index
label27_read_stream_id	int	0x0	Label27: Read Stream ID
label27_write_ssd	int	0x0	Label27: Write SDD or SSD_Index
label27_write_stream_id	int	0x0	Label27: Write Stream ID
label28_read_ssd	int	0x0	Label28: Read SDD or SSD_Index
label28_read_stream_id	int	0x0	Label28: Read Stream ID
label28_write_ssd	int	0x0	Label28: Write SDD or SSD_Index
label28_write_stream_id	int	0x0	Label28: Write Stream ID
label29_read_ssd	int	0x0	Label29: Read SDD or SSD_Index
label29_read_stream_id	int	0x0	Label29: Read Stream ID
label29_write_ssd	int	0x0	Label29: Write SDD or SSD_Index
label29_write_stream_id	int	0x0	Label29: Write Stream ID
label2_read_ssd	int	0x0	Label2: Read SDD or SSD_Index
label2_read_stream_id	int	0x0	Label2: Read Stream ID
label2_write_ssd	int	0x0	Label2: Write SDD or SSD_Index
label2_write_stream_id	int	0x0	Label2: Write Stream ID
label30_read_ssd	int	0x0	Label30: Read SDD or SSD_Index
label30_read_stream_id	int	0x0	Label30: Read Stream ID

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-386 Parameters for MMU_400 (continued)

Name	Type	Default value	Description
label30_write_ssd	int	0x0	Label30: Write SDD or SSD_Index
label30_write_stream_id	int	0x0	Label30: Write Stream ID
label31_read_ssd	int	0x0	Label31: Read SDD or SSD_Index
label31_read_stream_id	int	0x0	Label31: Read Stream ID
label31_write_ssd	int	0x0	Label31: Write SDD or SSD_Index
label31_write_stream_id	int	0x0	Label31: Write Stream ID
label3_read_ssd	int	0x0	Label3: Read SDD or SSD_Index
label3_read_stream_id	int	0x0	Label3: Read Stream ID
label3_write_ssd	int	0x0	Label3: Write SDD or SSD_Index
label3_write_stream_id	int	0x0	Label3: Write Stream ID
label4_read_ssd	int	0x0	Label4: Read SDD or SSD_Index
label4_read_stream_id	int	0x0	Label4: Read Stream ID
label4_write_ssd	int	0x0	Label4: Write SDD or SSD_Index
label4_write_stream_id	int	0x0	Label4: Write Stream ID
label5_read_ssd	int	0x0	Label5: Read SDD or SSD_Index
label5_read_stream_id	int	0x0	Label5: Read Stream ID
label5_write_ssd	int	0x0	Label5: Write SDD or SSD_Index
label5_write_stream_id	int	0x0	Label5: Write Stream ID
label6_read_ssd	int	0x0	Label6: Read SDD or SSD_Index
label6_read_stream_id	int	0x0	Label6: Read Stream ID
label6_write_ssd	int	0x0	Label6: Write SDD or SSD_Index
label6_write_stream_id	int	0x0	Label6: Write Stream ID
label7_read_ssd	int	0x0	Label7: Read SDD or SSD_Index
label7_read_stream_id	int	0x0	Label7: Read Stream ID
label7_write_ssd	int	0x0	Label7: Write SDD or SSD_Index
label7_write_stream_id	int	0x0	Label7: Write Stream ID
label8_read_ssd	int	0x0	Label8: Read SDD or SSD_Index
label8_read_stream_id	int	0x0	Label8: Read Stream ID
label8_write_ssd	int	0x0	Label8: Write SDD or SSD_Index
label8_write_stream_id	int	0x0	Label8: Write Stream ID
label9_read_ssd	int	0x0	Label9: Read SDD or SSD_Index
label9_read_stream_id	int	0x0	Label9: Read Stream ID
label9_write_ssd	int	0x0	Label9: Write SDD or SSD_Index
label9_write_stream_id	int	0x0	Label9: Write Stream ID
number_of_contexts	int	0x8	Number of context banks

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-386 Parameters for MMU_400 (continued)

Name	Type	Default value	Description
number_of_smrs	int	0x20	Number of stream match registers.
percent_tlbstatus_commits	int	0xa	Percentage of times that a poll of TLBSTATUS will commit the TLBI commands
prefetch_only_requests	int	0x0	Handle prefetch-only requests by:- 0 -- deny them 1 -- use debug table walks and TLB entries 2 -- treat them as normal transactions (dangerous)
programmable_non_secure_by_default_ssd_indices	string	""	Programmable SSD Indexes that are by default non-secure (e.g. 0, 6, 35-84).
programmable_secure_by_default_ssd_indices	string	""	Programmable SSD Indexes that are by default secure (e.g. 0, 6, 35-84).
ptw_has_separate_port	bool	0x1	Page Table Walks use pvbus_ptw_m
pvbus_m_is_ace_lite	bool	0x1	Is pvbus_m (the downstream port that translated transaction exit) ACE-Lite
pvbus_ptw_m_is_ace_lite	bool	0x1	Is pvbus_ptw_m (the downstream port that is used for walks if ptw_has_separate_port is true) ACE-Lite
stream_id_width	int	0x6	StreamID bit width
tlb_depth	int	0x40	TLB Depth (0 means 10000). The model will perform best with more TLB entries.
use_label_mapping	bool	0x1	Use label mapping
use_ssd_determination_table	bool	0x1	Use SSD Determination Table

Table 3-387 Parameters for MMU_400_BASE

Name	Type	Default value	Description
always_secure_ssd_indices	string	""	Non-programmable SSD Indexes that are always secure (e.g. 0, 6, 35-64).
cfg_cttw	bool	0x1	Perform coherent page table walks
dump_unpredictablity_in_user_flags	bool	0x0	Override the user flags to encode unpredictable information (validation only)
number_of_contexts	int	0x8	Number of context banks
number_of_smrs	int	0x10	Number of stream match registers.
percent_tlbstatus_commits	int	0xa	Percentage of times that a poll of TLBSTATUS will commit the TLBI commands
prefetch_only_requests	int	0x0	Handle prefetch-only requests by:- 0 -- deny them 1 -- use debug table walks and TLB entries 2 -- treat them as normal transactions (dangerous)
programmable_non_secure_by_default_ssd_indices	string	""	Programmable SSD Indexes that are by default non-secure (e.g. 0, 6, 35-84).

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-387 Parameters for MMU_400_BASE (continued)

Name	Type	Default value	Description
programmable_secure_by_default_ssd_indices	string	""	Programmable SSD Indexes that are by default secure (e.g. 0, 6, 35-84).
ptw_has_separate_port	bool	0x1	Page Table Walks use pvbus_ptw_m
pvbus_m_is_ace_lite	bool	0x1	Is pvbus_m (the downstream port that translated transaction exit) ACE-Lite
pvbus_ptw_m_is_ace_lite	bool	0x1	Is pvbus_ptw_m (the downstream port that is used for walks if ptw_has_separate_port is true) ACE-Lite
seed	int	0x12345678	Seed for SMMU
stream_id_width	int	0x6	StreamID bit width
tlb_depth	int	0x40	TLB Depth (0 means 10000). The model will perform best with more TLB entries.
use_ssd_determination_table	bool	0x1	Use SSD Determination Table

3.9.35 MMU_400_BASE

MMU-400 base component. This model is written in LISA+.

MMU_400_BASE contains the following CADI targets:

- MMU_400_BASE

MMU_400_BASE contains the following MTI components:

- *4.82 MMU_400_BASE - trace* on page 4-2547
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.106 PVBusMaster - trace* on page 4-2759
- *4.107 PVBusSlave - trace* on page 4-2760

Table 3-388 Ports

Name	Protocol	Type	Description
apb3_control_ns	PVBus	Slave	If the device has been configured with APB3 control ports then this is used to address the register file with non-secure accesses. If this is the case then the apb4_control port should not be used.
apb3_control_s	PVBus	Slave	If the device has been configured with APB3 control ports then this is used to address the register file with secure accesses. If this is the case then the apb4_control port should not be used.
apb4_control	PVBus	Slave	If the device has been configured with APB4 control ports then this port is used -- it carries the security world with the transaction itself. If this is the case then the apb3_control_s and apb3_control_ns should not be used.
cfg_cttw_in	*Signal* on page 2-85	Slave	The SoC supports coherent page walks, this is meant to be sampled at reset. However, in practice the model has to prevent the race condition between cfg_cttw being asserted at the same 'cycle' as negedge reset. Thus we actually only sample the signal on the first transaction to the SMMU or the first transition on this signal after reset. Thus in the model, we require that cfg_cttw be held for at least this period of time.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-388 Ports (continued)

Name	Protocol	Type	Description
cfg_flt_irpt_ns	*Signal* on page 2-85	Master	Non-secure Configuration Access Fault Interrupt In the SMMU Architecture this is called SMMU_NSgCfgIrpt.
cfg_flt_irpt_s	*Signal* on page 2-85	Master	Secure Configuration Access Fault Interrupt In the SMMU Architecture this is called SMMU_gCfgIrpt.
comb_irpt_ns	*Signal* on page 2-85	Master	"Non-secure combined interrupt" (cfg_flt_irpt_ns | glbl_flt_irpt_ns | cxt_irpt_ns)?
comb_irpt_s	*Signal* on page 2-85	Master	"Secure combined interrupt"
cxt_irpt_ns	*Signal* on page 2-85	Master	Non-secure context bank fault NOTE that there is only one context bank fault, despite there being potentially 8 contexts. As we are HW stage 2 only then we can't have any banks configured as secure (well if we do then we generate a global fault).
glbl_flt_irpt_ns	*Signal* on page 2-85	Master	Global non-secure Fault Interrupt In the SMMU Architecture this is called SMMU_NSgIrpt.
glbl_flt_irpt_s	*Signal* on page 2-85	Master	Global secure Fault Interrupt In the SMMU Architecture this is called SMMU_gIrpt.
identify	MMU_400_BASE_IDENTIFY	Master	This port is a special model port that is used to take a transaction and map it to an SSD/SSD_Index and StreamID.
priv_internals	MMU_400_Internals	Slave	For internal use only, please do not use.
pvbus_m	PVBus	Master	This downstream port is where the translated accesses from pvbus_s emerge. If page walks are configured to come out of this port, then they will come out with the with the same attributes as described for pvbus_ptw_m.
pvbus_ptw_m	PVBus	Master	This downstream port is where page table walk accesses come from. This is only used if configured to use a separate page table walk port. The MMU-400 will only obey DVM messages if configured to use this port. The page walks come out of this port with the following master_id and user_flags. master_id : 0xFFFFffff The user flags : user_flags[7:0] stage 1 context_id (or 0xFF if stage2 only) user_flags[15:8] stage 2 context_id (or 0xFF if stage 1 with stage 2 bypass) user_flags[18:16] stage 1 level user_flags[21:19] stage 2 level user_flags[23:22] stage 1 descriptor encoding (0=v7s, 1=v7l, 2=v8l, 3=none) user_flags[25:24] stage 2 descriptor encoding (0=v7s, 1=v7l, 2=v8l, 3=none) user_flags[31,30] adomain of the transaction NOTE that if the walk is being done for a stage 1 page walk descriptor fetch then the stage 1 level field will indicate that level. If the walk is being done for a stage 2 descriptor fetch, then the stage 2 level field will show that level. If the context-id for a stage is not valid (0xFF) then the 'level' information is 0x7.
pvbus_s	PVBus	Slave	This port is the upstream port of the device, addresses on the port are in the VA/IPA
reset_in	*Signal* on page 2-85	Slave	The reset pin.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-389 Parameters for MMU_400_BASE

Name	Type	Default value	Description
always_secure_ssd_indices	string	""	Non-programmable SSD Indexes that are always secure (e.g. 0, 6, 35-64).
cfg_cttw	bool	0x1	Perform coherent page table walks
dump_unpredictablity_in_user_flags	bool	0x0	Override the user flags to encode unpredictable information (validation only)
number_of_contexts	int	0x8	Number of context banks
number_of_smrs	int	0x10	Number of stream match registers.
percent_tlbstatus_commits	int	0xa	Percentage of times that a poll of TLBSTATUS will commit the TLBI commands
prefetch_only_requests	int	0x0	Handle prefetch-only requests by:- 0 -- deny them 1 -- use debug table walks and TLB entries 2 -- treat them as normal transactions (dangerous)
programmable_non_secure_by_default_ssd_indices	string	""	Programmable SSD Indexes that are by default non-secure (e.g. 0, 6, 35-84).
programmable_secure_by_default_ssd_indices	string	""	Programmable SSD Indexes that are by default secure (e.g. 0, 6, 35-84).
ptw_has_separate_port	bool	0x1	Page Table Walks use pvbus_ptw_m
pvbus_m_is_ace_lite	bool	0x1	Is pvbus_m (the downstream port that translated transaction exit) ACE-Lite
pvbus_ptw_m_is_ace_lite	bool	0x1	Is pvbus_ptw_m (the downstream port that is used for walks if ptw_has_separate_port is true) ACE-Lite
seed	int	0x12345678	Seed for SMMU
stream_id_width	int	0x6	StreamID bit width
tlb_depth	int	0x40	TLB Depth (0 means 10000). The model will perform best with more TLB entries.
use_ssd_determination_table	bool	0x1	Use SSD Determination Table

3.9.36 MMU_500

MMU-500 component. This model is written in LISA+.

MMU_500 contains the following CADI targets:

- MMU_500
- MMU_500_BASE

MMU_500 contains the following MTI components:

- *4.83 MMU_500_BASE - trace* on page 4-2576
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.106 PVBusMaster - trace* on page 4-2759
- *4.107 PVBusSlave - trace* on page 4-2760

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

About MMU_500

This is a model of a basic MMU-500. Set the version using the `version` parameter.

You cannot arbitrarily configure how you derive StreamIDs and SSD_Indexes from the transaction attributes.

This component has two label modes which you select using the parameter `use_label_mapping`:

- Set `use_label_mapping` to `true` if your upstream devices have labels in the top 16 bits of the transaction MasterID.

  Note

  The model does not have a concept of AXI-ID, but a transaction can have a MasterID set on it.

  Label your upstream components 0…*N* so that the parameters of this component can map those integers to StreamID and SSD_Index.
- Set `use_label_mapping` to `false` if the StreamID is encoded in the top 16 bits of the MasterID and the bottom 16 bits encode either the SSD_Index or the SSD state directly, depending on `use_ssd_determination_table`:
  - If `use_ssd_determination_table` is `true`, the bottom 16 bits of the MasterID encode the SSD_Index. They must be < 2^`ssd_index_width`.
  - If `use_ssd_determination_table` is `false`, the bottom 16 bits of the MasterID encode the SSD state directly, where zero is Secure and nonzero is Non-secure.

  Typically in hardware, a device emits different AXI-IDs, depending on what it is doing. In the model, MasterIDs are usually not diverse and a device might only emit one MasterID.

This component models the registers as follows:

- It models all architectural registers that the *Technical Reference Manual* (TRM) specifies, except that it does not model any of the performance registers.
- Unlike the MMU-400, MMU-500 does have an SMMU_STLBGSTATUS register because it has stage 1 and stage 2 support.
- The SMMU_NSACR is an alias of the Non-secure SMMU_ACR. This component models SMMU_ACR as RAZ/WI.
- The *ACR registers have IMP DEF contents. This component models only the PAGESIZE bit of the SACR, as non-RAZ/WI. It models no other IMP DEF registers.

Table 3-390 Ports

Name	Protocol	Type	Description
`cfg_cttw_in`	*Signal* on page 2-85	Slave	Enables coherent page table walks.
`comb_irpt_ns`	*Signal* on page 2-85	Master	Non-secure combined interrupt.
`comb_irpt_s`	*Signal* on page 2-85	Master	Secure combined interrupt.
`cxt_irpt[128]`	*Signal* on page 2-85	Master	Context interrupt.
`glbl_flt_irpt_ns`	*Signal* on page 2-85	Master	Global Non-secure fault interrupt.
`glbl_flt_irpt_s`	*Signal* on page 2-85	Master	Global Secure fault interrupt.
`priv_internals`	`MMU_500_Internals`	Slave	For internal use only, please do not use.
`pvbus_control_s`	`PVBus`	Slave	Provides memory-mapped read write access to the control registers of the module.
`pvbus_m[32]`	`PVBus`	Master	For all memory accesses. One for each Translation Buffer Unit (TBU).
`pvbus_ptw_m`	`PVBus`	Master	If ptw_has_separate_port is true, use for page table walks.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-390 Ports (continued)

Name	Protocol	Type	Description
pvbus_s[32]	PVBus	Slave	For transactions from PVBus master/decoder. One for each TBU.
reset_in	*Signal* on page 2-85	Slave	Reset signal.

Table 3-391 Parameters for MMU_500

Name	Type	Default value	Description
always_secure_ssd_indices	string	""	Non-programmable SSD Indexes that are always secure (e.g. 0, 6, 35-64).
cfg_cttw	bool	0x1	Perform coherent page table walks
dump_unpredictablity_in_user_flags	bool	0x0	Override the user flags to encode unpredictable information (validation only)
label0_read_ssd	int	0x0	Label0: Read SDD or SSD_Index
label0_read_stream_id	int	0x0	Label0: Read Stream ID
label0_write_ssd	int	0x0	Label0: Write SDD or SSD_Index
label0_write_stream_id	int	0x0	Label0: Write Stream ID
label10_read_ssd	int	0x0	Label10: Read SDD or SSD_Index
label10_read_stream_id	int	0x0	Label10: Read Stream ID
label10_write_ssd	int	0x0	Label10: Write SDD or SSD_Index
label10_write_stream_id	int	0x0	Label10: Write Stream ID
label11_read_ssd	int	0x0	Label11: Read SDD or SSD_Index
label11_read_stream_id	int	0x0	Label11: Read Stream ID
label11_write_ssd	int	0x0	Label11: Write SDD or SSD_Index
label11_write_stream_id	int	0x0	Label11: Write Stream ID
label12_read_ssd	int	0x0	Label12: Read SDD or SSD_Index
label12_read_stream_id	int	0x0	Label12: Read Stream ID
label12_write_ssd	int	0x0	Label12: Write SDD or SSD_Index
label12_write_stream_id	int	0x0	Label12: Write Stream ID
label13_read_ssd	int	0x0	Label13: Read SDD or SSD_Index
label13_read_stream_id	int	0x0	Label13: Read Stream ID
label13_write_ssd	int	0x0	Label13: Write SDD or SSD_Index
label13_write_stream_id	int	0x0	Label13: Write Stream ID
label14_read_ssd	int	0x0	Label14: Read SDD or SSD_Index
label14_read_stream_id	int	0x0	Label14: Read Stream ID
label14_write_ssd	int	0x0	Label14: Write SDD or SSD_Index
label14_write_stream_id	int	0x0	Label14: Write Stream ID
label15_read_ssd	int	0x0	Label15: Read SDD or SSD_Index
label15_read_stream_id	int	0x0	Label15: Read Stream ID

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-391 Parameters for MMU_500 (continued)

Name	Type	Default value	Description
label15_write_ssd	int	0x0	Label15: Write SDD or SSD_Index
label15_write_stream_id	int	0x0	Label15: Write Stream ID
label16_read_ssd	int	0x0	Label16: Read SDD or SSD_Index
label16_read_stream_id	int	0x0	Label16: Read Stream ID
label16_write_ssd	int	0x0	Label16: Write SDD or SSD_Index
label16_write_stream_id	int	0x0	Label16: Write Stream ID
label17_read_ssd	int	0x0	Label17: Read SDD or SSD_Index
label17_read_stream_id	int	0x0	Label17: Read Stream ID
label17_write_ssd	int	0x0	Label17: Write SDD or SSD_Index
label17_write_stream_id	int	0x0	Label17: Write Stream ID
label18_read_ssd	int	0x0	Label18: Read SDD or SSD_Index
label18_read_stream_id	int	0x0	Label18: Read Stream ID
label18_write_ssd	int	0x0	Label18: Write SDD or SSD_Index
label18_write_stream_id	int	0x0	Label18: Write Stream ID
label19_read_ssd	int	0x0	Label19: Read SDD or SSD_Index
label19_read_stream_id	int	0x0	Label19: Read Stream ID
label19_write_ssd	int	0x0	Label19: Write SDD or SSD_Index
label19_write_stream_id	int	0x0	Label19: Write Stream ID
label1_read_ssd	int	0x0	Label1: Read SDD or SSD_Index
label1_read_stream_id	int	0x0	Label1: Read Stream ID
label1_write_ssd	int	0x0	Label1: Write SDD or SSD_Index
label1_write_stream_id	int	0x0	Label1: Write Stream ID
label20_read_ssd	int	0x0	Label20: Read SDD or SSD_Index
label20_read_stream_id	int	0x0	Label20: Read Stream ID
label20_write_ssd	int	0x0	Label20: Write SDD or SSD_Index
label20_write_stream_id	int	0x0	Label20: Write Stream ID
label21_read_ssd	int	0x0	Label21: Read SDD or SSD_Index
label21_read_stream_id	int	0x0	Label21: Read Stream ID
label21_write_ssd	int	0x0	Label21: Write SDD or SSD_Index
label21_write_stream_id	int	0x0	Label21: Write Stream ID
label22_read_ssd	int	0x0	Label22: Read SDD or SSD_Index
label22_read_stream_id	int	0x0	Label22: Read Stream ID
label22_write_ssd	int	0x0	Label22: Write SDD or SSD_Index
label22_write_stream_id	int	0x0	Label22: Write Stream ID
label23_read_ssd	int	0x0	Label23: Read SDD or SSD_Index

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-391 Parameters for MMU_500 (continued)

Name	Type	Default value	Description
label23_read_stream_id	int	0x0	Label23: Read Stream ID
label23_write_ssd	int	0x0	Label23: Write SDD or SSD_Index
label23_write_stream_id	int	0x0	Label23: Write Stream ID
label24_read_ssd	int	0x0	Label24: Read SDD or SSD_Index
label24_read_stream_id	int	0x0	Label24: Read Stream ID
label24_write_ssd	int	0x0	Label24: Write SDD or SSD_Index
label24_write_stream_id	int	0x0	Label24: Write Stream ID
label25_read_ssd	int	0x0	Label25: Read SDD or SSD_Index
label25_read_stream_id	int	0x0	Label25: Read Stream ID
label25_write_ssd	int	0x0	Label25: Write SDD or SSD_Index
label25_write_stream_id	int	0x0	Label25: Write Stream ID
label26_read_ssd	int	0x0	Label26: Read SDD or SSD_Index
label26_read_stream_id	int	0x0	Label26: Read Stream ID
label26_write_ssd	int	0x0	Label26: Write SDD or SSD_Index
label26_write_stream_id	int	0x0	Label26: Write Stream ID
label27_read_ssd	int	0x0	Label27: Read SDD or SSD_Index
label27_read_stream_id	int	0x0	Label27: Read Stream ID
label27_write_ssd	int	0x0	Label27: Write SDD or SSD_Index
label27_write_stream_id	int	0x0	Label27: Write Stream ID
label28_read_ssd	int	0x0	Label28: Read SDD or SSD_Index
label28_read_stream_id	int	0x0	Label28: Read Stream ID
label28_write_ssd	int	0x0	Label28: Write SDD or SSD_Index
label28_write_stream_id	int	0x0	Label28: Write Stream ID
label29_read_ssd	int	0x0	Label29: Read SDD or SSD_Index
label29_read_stream_id	int	0x0	Label29: Read Stream ID
label29_write_ssd	int	0x0	Label29: Write SDD or SSD_Index
label29_write_stream_id	int	0x0	Label29: Write Stream ID
label2_read_ssd	int	0x0	Label2: Read SDD or SSD_Index
label2_read_stream_id	int	0x0	Label2: Read Stream ID
label2_write_ssd	int	0x0	Label2: Write SDD or SSD_Index
label2_write_stream_id	int	0x0	Label2: Write Stream ID
label30_read_ssd	int	0x0	Label30: Read SDD or SSD_Index
label30_read_stream_id	int	0x0	Label30: Read Stream ID
label30_write_ssd	int	0x0	Label30: Write SDD or SSD_Index
label30_write_stream_id	int	0x0	Label30: Write Stream ID

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-391 Parameters for MMU_500 (continued)

Name	Type	Default value	Description
label31_read_ssd	int	0x0	Label31: Read SDD or SSD_Index
label31_read_stream_id	int	0x0	Label31: Read Stream ID
label31_write_ssd	int	0x0	Label31: Write SDD or SSD_Index
label31_write_stream_id	int	0x0	Label31: Write Stream ID
label3_read_ssd	int	0x0	Label3: Read SDD or SSD_Index
label3_read_stream_id	int	0x0	Label3: Read Stream ID
label3_write_ssd	int	0x0	Label3: Write SDD or SSD_Index
label3_write_stream_id	int	0x0	Label3: Write Stream ID
label4_read_ssd	int	0x0	Label4: Read SDD or SSD_Index
label4_read_stream_id	int	0x0	Label4: Read Stream ID
label4_write_ssd	int	0x0	Label4: Write SDD or SSD_Index
label4_write_stream_id	int	0x0	Label4: Write Stream ID
label5_read_ssd	int	0x0	Label5: Read SDD or SSD_Index
label5_read_stream_id	int	0x0	Label5: Read Stream ID
label5_write_ssd	int	0x0	Label5: Write SDD or SSD_Index
label5_write_stream_id	int	0x0	Label5: Write Stream ID
label6_read_ssd	int	0x0	Label6: Read SDD or SSD_Index
label6_read_stream_id	int	0x0	Label6: Read Stream ID
label6_write_ssd	int	0x0	Label6: Write SDD or SSD_Index
label6_write_stream_id	int	0x0	Label6: Write Stream ID
label7_read_ssd	int	0x0	Label7: Read SDD or SSD_Index
label7_read_stream_id	int	0x0	Label7: Read Stream ID
label7_write_ssd	int	0x0	Label7: Write SDD or SSD_Index
label7_write_stream_id	int	0x0	Label7: Write Stream ID
label8_read_ssd	int	0x0	Label8: Read SDD or SSD_Index
label8_read_stream_id	int	0x0	Label8: Read Stream ID
label8_write_ssd	int	0x0	Label8: Write SDD or SSD_Index
label8_write_stream_id	int	0x0	Label8: Write Stream ID
label9_read_ssd	int	0x0	Label9: Read SDD or SSD_Index
label9_read_stream_id	int	0x0	Label9: Read Stream ID
label9_write_ssd	int	0x0	Label9: Write SDD or SSD_Index
label9_write_stream_id	int	0x0	Label9: Write Stream ID
number_of_contexts	int	0x8	Number of context banks
number_of_smrs	int	0x20	Number of stream match registers.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-391 Parameters for MMU_500 (continued)

Name	Type	Default value	Description
percent_tlbstatus_commits	int	0xa	Percentage of times that a poll of TLBSTATUS will commit the TLBI commands
prefetch_only_requests	int	0x0	Handle prefetch-only requests by:- 0 -- deny them 1 -- use debug table walks and TLB entries 2 -- treat them as normal transactions (dangerous)
programmable_non_secure_by_default_ssd_indices	string	""	Programmable SSD Indexes that are by default non-secure (e.g. 0, 6, 35-84).
programmable_secure_by_default_ssd_indices	string	""	Programmable SSD Indexes that are by default secure (e.g. 0, 6, 35-84).
ptw_has_separate_port	bool	0x1	Page Table Walks use pvbus_ptw_m
supports_nested_translations	bool	0x1	Supports nested translations (stage 1 + stage 2)
tlb_depth	int	0x800	TLB Depth (0 means 10000). The model will perform best with more TLB entries.
use_label_mapping	bool	0x1	Use label mapping
use_ssd_determination_table	bool	0x1	Use SSD Determination Table
version	string	"EAC"	Version of the RTL that the model represents. Valid values are LACr1 and EAC.

Table 3-392 Parameters for MMU_500_BASE

Name	Type	Default value	Description
PRIVATE_PARAMETER_personality	string	""	The personality to use (affects ID codes and various imp def features).
PRIVATE_PARAMETER_seed	int	0x12345678	Seed for randomised SMMU implementation defined behaviour
PRIVATE_PARAMETER_validation_mode	int	0x0	Internal validation mode.
always_secure_ssd_indices	string	""	Non-programmable SSD Indexes that are always secure (e.g. 0, 6, 35-64).
cfg_cttw	bool	0x1	Perform coherent page table walks
dump_unpredictablity_in_user_flags	bool	0x0	Override the user flags to encode unpredictable information (validation only)
number_of_contexts	int	0x8	Number of context banks
number_of_smrs	int	0x10	Number of stream match registers.
percent_tlbstatus_commits	int	0xa	Percentage of times that a poll of TLBSTATUS will commit the TLBI commands
prefetch_only_requests	int	0x0	Handle prefetch-only requests by:- 0 -- deny them 1 -- use debug table walks and TLB entries 2 -- treat them as normal transactions (dangerous)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-392 Parameters for MMU_500_BASE (continued)

Name	Type	Default value	Description
programmable_non_secure_by_default_ssd_indices	string	""	Programmable SSD Indexes that are by default non-secure (e.g. 0, 6, 35-84).
programmable_secure_by_default_ssd_indices	string	""	Programmable SSD Indexes that are by default secure (e.g. 0, 6, 35-84).
ptw_has_separate_port	bool	0x1	Page Table Walks use pvbus_ptw_m (or uses pvbus_m[0])
supports_nested_translations	bool	0x1	Supports nested translations (stage 1 + stage 2)
tlb_depth	int	0x800	TLB Depth (0 means 10000). The model will perform best with more TLB entries.
use_ssd_determination_table	bool	0x1	Use SSD Determination Table
version	string	"EAC"	Version of the RTL that the model represents. Valid values are LACr1 and EAC.

3.9.37 MMU_500_BASE

MMU-500 base component. This model is written in LISA+.

MMU_500_BASE contains the following CADI targets:

- MMU_500_BASE

MMU_500_BASE contains the following MTI components:

- *4.83 MMU_500_BASE - trace* on page 4-2576
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.106 PVBusMaster - trace* on page 4-2759
- *4.107 PVBusSlave - trace* on page 4-2760

Table 3-393 Ports

Name	Protocol	Type	Description
cfg_cttw_in	*Signal* on page 2-85	Slave	The SoC supports coherent page walks, this is meant to be sampled at reset. However, in practice the model has to prevent the race condition between cfg_cttw being asserted at the same 'cycle' as negedge reset. Thus we actually only sample the signal on the first transaction to the SMMU or the first transition on this signal after reset. Thus in the model, we require that cfg_cttw be held for at least this period of time.
comb_irpt_ns	*Signal* on page 2-85	Master	"Non-secure combined interrupt"
comb_irpt_s	*Signal* on page 2-85	Master	"Secure combined interrupt"
cxt_irpt[128]	*Signal* on page 2-85	Master	Non-secure context bank fault.
glbl_flt_irpt_ns	*Signal* on page 2-85	Master	Global non-secure Fault Interrupt In the SMMU Architecture this is called SMMU_NSgIrpt.
glbl_flt_irpt_s	*Signal* on page 2-85	Master	Global secure Fault Interrupt In the SMMU Architecture this is called SMMU_gIrpt.
identify	MMU_500_BASE_IDENTIFY	Master	This port is a special model port that is used to take a transaction and map it to an SSD/SSD_Index and StreamID.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-393 Ports (continued)

Name	Protocol	Type	Description
priv_internals	MMU_500_Internals	Slave	For internal use only, please do not use.
pvbus_control_s	PVBus	Slave	The register port of the device is AXI.
pvbus_m[32]	PVBus	Master	This downstream port is where the translated accesses from pvbus_s emerge. See notes for pvbus_s[] as well. If the Page Table Walk (PTW) does not have a separate port then PTW accesses will emerge at port 0 with the same attributes as described in pvbus_ptw_m.
pvbus_ptw_m	PVBus	Master	This downstream port is where page table walk accesses come from. This is only used if configured to use a separate page table walk port. The MMU-500 will only obey DVM messages if configured to use this port. The page walks come out of this port with the following master_id and user_flags. master_id : 0xFFFFffff The user flags : user_flags[7:0] stage 1 context_id (or 0xFF if stage2 only) user_flags[15:8] stage 2 context_id (or 0xFF if stage 1 with stage 2 bypass) user_flags[18:16] stage 1 level user_flags[21:19] stage 2 level user_flags[31,30] adomain of the transaction NOTE that if the walk is being done for a stage 1 page walk descriptor fetch then the stage 1 level field will indicate that level. If the walk is being done for a stage 2 descriptor fetch, then the stage 2 level field will show that level. If the context-id for a stage is not valid (0xFF) then the 'level' information is 0x7.
pvbus_s[32]	PVBus	Slave	This port is the upstream port of the device, addresses on the port are in the VA/IPA Each TBU in the design is represented by a pair of pvbus_s[tbu_id] and pvbus_m[tbu_id]. That is transactions that go into pvbus_s[tbu_id] will emerge at pvbus_m[tbu_id]. The port index that a transaction comes in on is the tbu_number_ parameter to the MMU_500_BASE_IDENTIFY::identify() function. The identify() function must use all the information it is given by the parameters to map to the architectural concepts of StreamID and SSD_Index/SSD. How it does this is IMPLEMENTATION DEFINED and depends on the topology of the SoC and the masters upstream of the TBUs.
reset_in	*Signal* on page 2-85	Slave	The reset pin.

Table 3-394 Parameters for MMU_500_BASE

Name	Type	Default value	Description
PRIVATE_PARAMETER_personality	string	""	The personality to use (affects ID codes and various imp def features).
PRIVATE_PARAMETER_seed	int	0x12345678	Seed for randomised SMMU implementation defined behaviour
PRIVATE_PARAMETER_validation_mode	int	0x0	Internal validation mode.
always_secure_ssd_indices	string	""	Non-programmable SSD Indexes that are always secure (e.g. 0, 6, 35-64).
cfg_cttw	bool	0x1	Perform coherent page table walks
dump_unpredictablity_in_user_flags	bool	0x0	Override the user flags to encode unpredictable information (validation only)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-394 Parameters for MMU_500_BASE (continued)

Name	Type	Default value	Description
number_of_contexts	int	0x8	Number of context banks
number_of_smrs	int	0x10	Number of stream match registers.
percent_tlbstatus_commits	int	0xa	Percentage of times that a poll of TLBSTATUS will commit the TLBI commands
prefetch_only_requests	int	0x0	Handle prefetch-only requests by:- 0 -- deny them 1 -- use debug table walks and TLB entries 2 -- treat them as normal transactions (dangerous)
programmable_non_secure_by_default_ssd_indices	string	""	Programmable SSD Indexes that are by default non-secure (e.g. 0, 6, 35-84).
programmable_secure_by_default_ssd_indices	string	""	Programmable SSD Indexes that are by default secure (e.g. 0, 6, 35-84).
ptw_has_separate_port	bool	0x1	Page Table Walks use pvbus_ptw_m (or uses pvbus_m[0])
supports_nested_translations	bool	0x1	Supports nested translations (stage 1 + stage 2)
tlb_depth	int	0x800	TLB Depth (0 means 10000). The model will perform best with more TLB entries.
use_ssd_determination_table	bool	0x1	Use SSD Determination Table
version	string	"EAC"	Version of the RTL that the model represents. Valid values are LACr1 and EAC.

3.9.38 MMU_600

This model is written in C++.

MMU_600 contains the following CADI targets:

- MMU_600

MMU_600 contains the following MTI components:

- *4.84 MMU_600 - trace* on page 4-2605
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760

About MMU_600 and MMU_700

MMU_600 and MMU_700 are SMMUv3-compliant devices and are used for I/O virtualization of devices.

The hardware is a distributed SMMU. It consists of the following:

- A single Translation Control Unit (TCU), which has a port for the programming interface of the SMMU, receives DVM messages, and does all the page walking and queue manipulation, for example.
- One or more Translation Bus Units (TBUs), which translate transactions from upstream client devices into downstream transactions.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

- Zero or more connections to PCIe Root Complexes (PCIe-RCs). A maximum of 62 TBUs and PCIe-RCs can be attached.
- An interconnect that connects the TBUs and PCIe-RCs to the TCU.

A PCIe-RC has one connection to the TCU to make ATS requests but the PCIe-RC uses one or more TBUs to transform the transactions and pass them to the memory system. In the model, the TBUs used are listed in the parameter `list_of_pcie_mode`. The SMMU does not know which TBUs a particular PCIe-RC is attached to.

The TBU ORs a value into each StreamID that it receives. In the model, this is configured by the following parameters:

- `list_of_s_sid_high_at_bitpos0`
- `list_of_ns_sid_high_at_bitpos0`

The TCU, TBU, and the interconnect are all represented by this single model component.

In the model, the `tbs_pvbus_s[i]` and `tbm_pvbus_m[i]` port pair represent a TBU `i`, or `tbs_pvbus_s[i]` represents an incoming connection for a PCIe-RC. The corresponding reverse connection from the TCU to the PCIe-RC is by a special bus called `pvbus_id_routed_m` that is used to transport ATC Invalidates to the PCIe-RC.

To reduce system construction complexity, the `tbs_pvbus_s[i]` and `tbm_pvbus_m[i]` pair also acts as a TBU so that the PCIe-RC does not need to separate its normal transactions and its ATS requests.

However, ATC Invalidates are only sent to a port which appears in `list_of_pcie_rc`. It should be uniquely decoded to a single port based on `list_of_ns_sid_high_at_bitpos0`, and ATC Invalidates must be routed to the correct PCIe subsystem to invalidate the cache of ATS Responses in the subsystem. Therefore all TBUs that a PCIe-RC uses must have a unique reverse mapping from stream id to port.

Note

A bad configuration renders the model inactive.

Some configuration can be adjusted by configuration pins. These are only sampled at the negative edge of the reset pin. If you want to use these pins, then you must drive them before sending a negative edge on the reset pin. During simulation_reset, the component driving them must also drive this transition again.

The pin `sup_oas` is not supported, instead it is a parameter, as it is assumed that it would be tied to a fixed value in any specific platform.

Debug reads to the registers do not disturb the state.

Writes to registers with Update flags, including debug writes, are ignored if the Update flag is already set to one.

Debug and real accesses to the registers must be 32 bits or 64 bits.

MSIs are issued on the `qtw_pvbus_m` port using attributes that are determined by the parameter `msi_attribute_transform`, while Event queue writes are always issued with `ExtendedID=0, UserFlags=0, MasterID=0xFFFFffff`. In the hardware, there is no way of distinguishing Event queue writes from MSI writes. However, this provides a mechanism for the model system to distinguish them.

The hardware only has a single cacheability attribute for input transactions, but `PVBus` transports have both inner and outer cacheability.

For non-PCIe-mode TBUs, whose index does not appear in `list_of_pcie_mode` or `list_of_pcie_rc`, for non-cache maintenance operations:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

- If the input attribute is any type of device then it is well defined as being outer-shared and Device-nGnRnE or Device-nGnRE. There is no support for Gathering or Reordering.
- If the outer cacheable input attribute is normal, and it is Write-back, this is converted to Inner Write-back Outer Write-back (iWB-oWB) with the desired shareability. No Transient hint is supported and is always treated as non-transient.
- This leaves all other normal memory types that are mapped to Inner Normal Non-cacheable, Outer Normal Non-cacheable outer shared (iNC-oNC-osh).

Therefore, the upstream devices must present the cacheability in the *outer* cacheability attribute on `PVBus` if it is cacheable. If it is a device type then both the inner and outer attributes must be set to the same value. If it is iNC-oNC-osh, then it must be presented as such.

For PCIe-mode TBUs, that is, whose index appears in `list_of_pcie_mode` or `list_of_pcie_rc`:

- Input transactions are from PCIe and the only indication of the memory type is in the NoSnoop bit of the transaction. No shareability is transported. NoSnoop is interpreted as iNC-oNC-osh. ! NoSnoop is interpreted as iWB-oWB-ish (note Inner Shareable).
- If a NoSnoop transaction has an attribute transform applied to it and the result of the transform is weaker than iNC-oNC-osh, it is forced to iNC-oNC-osh. For example, if a NoSnoop transaction uses a page table and is transformed to iWB-oWB-nsh then it is forced to iNC-oNC-osh. However, if the page tables transformed it to a device type, then as all device types are stronger than iNC-oNC-osh, it exits the SMMU as the device type.
- In the model, transactions are classified by their incoming memory attributes as to whether they are NoSnoop or not and then are normalized appropriately:
  - iWB-oWB-any-shareability are interpreted as ! NoSnoop, therefore are normalized to iWB-oWB-ish.
  - Anything else is considered NoSnoop, and therefore is normalized to iNC-oNC-osh.
- Translated accesses also have the same interpretation to determine NoSnoop and how they are normalized. Therefore they could enter the system with different attributes to if they entered the SMMU as Untranslated Accesses and were translated by the normal translation process, before exiting downstream of the SMMU and entering the system
- It is expected that there is a component downstream of the SMMU that is aware of the system address map and will override the memory type to `device` for any transaction that accesses a peripheral.

The hardware has a single cacheability on input, and, for transactions that are neither cache-maintenance operations nor PCIe transactions, it normalizes the input to an architectural form before performing the SMMUv3 architectural transform:

- Any device type is left untouched. The input can only represent Device-nGnRE and Device-nGnRnE.
- If the input is Write-back (WB), it is normalized to iWB-oWB with the incoming shareability.
- If the input is anything else, it is normalized to iNC-oNC-osh.

The model accepts full architectural attributes of two levels of cacheability and so has to decide how to interpret this in terms of the hardware. For transactions that are not cache maintenance operations, the model replicates the outer attribute into the inner attribute and then performs the normalization that the hardware does.

The hardware normalizes the architectural output attributes and outputs a single level of cacheability and a user flag (OC) specifying if the architectural attributes were cacheable in the outer cacheable domain. If the transaction is classified as a PCIe transaction, the NoSnoop transform previously described is applied. That is, if the original transaction was NoSnoop then any weaker memory type is strengthened to iNC-oNC-osh, and the following transform is applied:

```
if      iWB-oWB-nsh/ish/osh     then output WB-nsh/ish/osh, OC = 1
else if i(NC/WT/WB)-o(WB/WT)    then output NC-Sys,         OC = 1
else if i(NC/WT/WB)-oNC         then output NC-Sys,         OC = 0
else if Device-(GRE/nGRE/nGnRE) then output DV-Sys,         OC = 0
else                                 output SO-Sys,         OC = 0
```


Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

The model only normalizes according to PCIe but otherwise leaves the architectural attributes intact on the output bus.

Important

The model deals with groups of transactions with the same attributes and a similar range of addresses. The mapping that is used is stored by the bus infrastructure and is used for subsequent sufficiently similar transactions without needing the intervention of the SMMU model, and therefore are not traced.

Model limitations

- The Performance Monitoring Unit (PMU) has limited functionality. It is intended for demonstration purposes only and does not implement all architecturally required events. It does not implement the `pmusnapshot_ack` or `pmusnapshot_req` interface.
- The model does not implement:
  - RAS.
  - Power control.
  - `sup_oas`, which controls the OAS of the SMMU. This is expected to be constant for a system.
  - The SYSCO interface.
  - The Low-Power Interface
- Cache maintenance operations cannot be inserted into the TBU ports of the SMMU.
- `PVBus` has no representation of the cache stash operations, so they are not supported.
- `TCU_CFG.XLATE_SLOTS` is fixed at 512.
- `TCU_STATUS.GNT_XLATE_SLOTS` always reads as 512.

Table 3-395 Ports

Name	Protocol	Type	Description
`clk_in`	`ClockSignal`	Slave	Clock signal (in RTL aclk) This is a clock time-base used by the TCU to spread some of its processing over time, if enabled by the wait_* parameters. The clock must always be connected.
`cmd_sync_irpt_ns`	*Signal* on page 2-85	Master	Pulsed interrupt output signal for non-secure CMD_SYNC having a completion signal of SIG_IRQ.
`cmd_sync_irpt_s`	*Signal* on page 2-85	Master	Pulsed interrupt output signal for the PRI queue becoming non-empty. Pulsed interrupt output signal for secure CMD_SYNC having a completion signal of SIG_IRQ.
`event_q_irpt_ns`	*Signal* on page 2-85	Master	Pulsed interrupt output signal for the non-secure event queue becoming non-empty.
`event_q_irpt_s`	*Signal* on page 2-85	Master	Pulsed interrupt output signal for the secure event queue becoming non-empty.
`evento`	*Signal* on page 2-85	Master	Event signal
`global_irpt_ns`	*Signal* on page 2-85	Master	Pulsed interrupt output signal for non-secure SMMU_GERROR(N) signalling an error.
`global_irpt_s`	*Signal* on page 2-85	Master	Pulsed interrupt output signal for secure SMMU_S_GERROR(N) signalling an error.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-395 Ports (continued)

Name	Protocol	Type	Description
identify	*SMMUv3AEMIdentifyProtocol* on page 2-78	Master	Map the transaction to the tuple (StreamID, SubStreamID, SubStreamIDValid, SSD) The StreamID that is produced by the implementation of this protocol is not the final StreamID. The final StreamID is produced by using the list_of_ns_sid_high_at_bitpos0/ list_of_s_sid_high_at_bitpos0 parameter to map the StreamID based on the upstream port index. Also see the parameter howto_identify which can replace the functionality of this port under certain circumstances.
prog_pvbus_s	PVBus	Slave	Register slave port (in RTL PROG)
pvbus_id_routed_m[62]	PVBus	Master	This is a special "id-routed" port for transmitting ATC invalidates upstream into the PCIe EndPoints, it is not a normal bus. The FastModels ATC Invalidate and PRI Response protocol specifies how to route and deal with this port.
qtw_pvbus_m	PVBus	Master	Master port used for Table Walks, MSIs and Queue access when separate_tw_msi_qs_port==true (in RTL QTW)
sec_override	*Signal* on page 2-85	Slave	Allow certain registers to be accessible to non-secure accesses from reset, as described in the TCU_SCR register.
sup_btm	*Signal* on page 2-85	Slave	System supports BTM and will be reflected in the IDR registers. This signal can override the value set by the parameters configuring the IDR registers. If BTM (Broadcast Table Maintenance) is not supported then DVM messages will be ignored.
sup_cohacc	*Signal* on page 2-85	Slave	System supports COHACC and will be reflected in the IDR registers.
sup_sev	*Signal* on page 2-85	Slave	System supports SEV and will be reflected in the IDR registers. This signal can override the value set by the parameters configuring the IDR registers.
tbm_pvbus_m[62]	PVBus	Master	The TBU master ports that carry transactions that have been translated from the correspondingly numbered tbs_pvbus_s[] port.
tbs_pvbus_s[62]	PVBus	Slave	The TBU slave ports that receive transactions to be translated. They will exit the SMMU through the same numbered pvbus_m[] port.
tbu_pmu_irpt[62]	*Signal* on page 2-85	Master	TBU Performance Monitoring Unit interrupt, one per TBU.
tbu_pmusnapshot_ack[62]	*Signal* on page 2-85	Master	PMU snapshot interface for the TBU, ack a snapshot.
tbu_pmusnapshot_req[62]	*Signal* on page 2-85	Slave	PMU snapshot interface for the TBU, request a snapshot.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-395 Ports (continued)

Name	Protocol	Type	Description
tbu_ras_irpt[62]	*Signal* on page 2-85	Master	The RAS interrupt pin for errors detected in the TBUs.
tbu_reset_in[62]	*Signal* on page 2-85	Slave	Reset signals The TBUs can have independent reset signals. Each signal tbu_reset_in[n] corresponds to the TBU using tbs_pvbus_s[n]/ tbs_pvbus_s[n] pair. If the SMMU receives a transaction whilst the TBU is expected to be in reset then it will complain using the ArchMsg.Warning.warning trace source. Those tbu_reset_in that correspond to a PCIe-RC connection can be connected to monitor the PCIe-RC's reset signal. If it receives an ATS request when in reset then it will complain in a similar way. You must connect these pins if you wish the TCU_NODE_STATUS for the nodes to be accurate (including any connected to the PCIe-RC).
tcu_pmu_irpt	*Signal* on page 2-85	Master	TCU Performance Monitoring Unit interrupt
tcu_pmusnapshot_ack	*Signal* on page 2-85	Master	PMU snapshot interface for the TCU, ack a snapshot.
tcu_pmusnapshot_req	*Signal* on page 2-85	Slave	PMU snapshot interface This is a four-phase handshake to have the corresponding PMCG perform a capture of the current counter values. PMU snapshot interface for the TCU, request a snapshot.
tcu_ras_irpt	*Signal* on page 2-85	Master	The RAS interrupt pin for errors detected in the TCU.
tcu_reset_in	*Signal* on page 2-85	Slave	The reset signal to the TCU interface.

Table 3-396 Parameters for MMU_600

Name	Type	Default value	Description
all_error_messages_through_trace	bool	0x0	Some conditions in the SMMU are so strange that the software programming the SMMU has done something wrong. At this point messages are output to either ArchMsg.Error.* or ArchMsg.Warning.* or to the error stream of the simulator. Outputting to the error stream of the simulator may cause it to return with a non-zero exit status. If you set this option to true then instead of using the error stream of the simulator it will always use a trace stream allowing the simulation to exit with a zero exit status.
behaviour_of_sampled_at_reset_signals	int	0x0	Some configuration signals into the SMMU are sampled on negedge of reset. However, it can sometimes be hard to arrange to drive a configuration pin before the negedge of reset. The configuration pins are sampled: 0 -- at negedge reset. 1 -- at negedge reset, but if a later change occurs at the same simulated time, and no transactions have occurred, then they will be resampled and the SMMU reset again.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-396 Parameters for MMU_600 (continued)

Name	Type	Default value	Description
cmdq_max_number_of_commands_to_buffer	int	0xa	The command queues can buffer fetched commands before issuing them. This parameter is roughly the maximum number of commands to do this for. The programmer visible effects are that just because the CONS pointer shows a command has been _consumed_ does not necessarily mean that it has been issued (and completed). Higher values will accentuate this effect.
enable_device_id_checks	bool	0x1	If this parameter is true then the DeviceIDs seen by the GIC are: * for client devices: DeviceID = StreamID + translated_device_id_base * for SMMU-generated MSIs: smmu_msi_device_id This parameter enables two checks: * If the DeviceID is used in the output_attribute_transform then if it overflows 32 bits then the model will warn. If the DeviceID is not used then it is assumed that the external agent that forms the DeviceID will warn if it overflows. * If the SMMU supports MSIs, then the model will check that the GIC will be able to distinguish an MSI generated by the SMMU from one generated by a client device. As the exact mechanism to determine the DeviceID is in the system and not necessarily under control of the SMMU then you can disable these warnings using this parameter. See also the parameters: output_attribute_transform and msi_attribute_transform.
howto_identify	string	"use-identify"	If 'use-identify' then will use the 'identify' port to determine the SSD, StreamID, SubStreamID of a transaction. Otherwise, this string extracts them from the transaction's attributes. Examples:- SEC_SID=ExtendedID[63], SSV=ExtendedID[62], SubstreamID=ExtendedID[51:32], StreamID=ExtendedID[31:0] nSEC_SID=ExtendedID[63], StreamID=ExtendedID[55:24], nSSV=ExtendedID[20], SubstreamID=ExtendedID[19:0] SEC_SID is one bit wide, true if StreamID is secure. SSV is one bit wide, true if SubstreamID is valid. The alternative symbols nSEC_SID and nSSV have the negative logic to SEC_SID and SSV. You must not use the negative and positive logic for the same symbol at the same time. However, just because you use negative logic for one symbol does not force you to use negative logic for the other. SubstreamID is 20 bits wide and StreamID is 32 bits. More complex examples: StreamID[31:24]=0, StreamID[23:0]=ExtendedID[23:0], SSV=1[0], ... Available attributes: ExtendedID, MasterID, UserFlags
list_of_ns_sid_high_at_bitpos0	string	""	A comma-separated list of values to bitwise OR into each Non-secure StreamID for each TBU/Node. Bit 0 of the value corresponds to bit 0 of the StreamID. Each TBU that is connected to a PCIe subsystem must serve a unique contiguous subset of StreamIDs as determined by their top bits. This is used in order to know which port to route ATC Invalidates to the PCIe subsystems.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-396 Parameters for MMU_600 (continued)

Name	Type	Default value	Description
list_of_pcie_mode	string	""	A comma-separated list of ranges of ports that represent TBUs that are attached to PCIe Root Complexes (PCIe-RC). A single PCIe-RC might use several TBUs and stripe accesses across them. The attributes handling for these TBUs are slightly different in that if the PCIe transaction is NoSnoop and the output attributes of the translation would be weaker than iNC-oNC-osh then the output is forced to iNC-oNC-osh. iNC-oNC-osh == "inner normal non-cacheable, out normal non-cacheable, outer shared"
list_of_pcie_rc	string	""	This is a list of ports that are connected to PCIe Root Complex (PCIe-RC) by a protocol called DTI-ATS. This port is used to transport ATS and PRI Requests to the SMMU from the PCIe-RC. In the real hardware, then the PCIe-RC uses this port for ATS/PRI, and then the actual transactions go through separate TBUs. In the model, then this port can accept actual transactions as well. However, in the model, then the ATC Invalidates and the PRI Responses need to be transferred over the corresponding pvbus_id_routed_m port as DTI-ATS is bidirectional, but PVBus is not.
list_of_s_sid_high_at_bitpos0	string	""	A comma-separated list of values to bitwise OR into each Secure StreamID for each TBU/Node. Bit 0 of the value corresponds to bit 0 of the StreamID.
msi_attribute_transform	string	"ExtendedID[31:0]=smmu_msi_device_id, MasterID=0xFFFFffff"	This parameter determines the MSI output attributes of SMMU-generated MSIs: * MasterID * UserFlags * ExtendedID of the transaction to use. This is a set of comma separated transforms on the output attributes. The right hand side is: * a numeric literal (or a slice of a literal) * the parameter smmu_msi_device_id * the symbol 'interrupt_kind' * 0/1 -- EVENTQ Secure/Non-secure * 2 -- PRIQ * 3/4 -- CMD_SYNC Secure/Non-secure * 5/6 -- GERROR Secure/Non-secure * 7/8 -- PMCG Secure/Non-Secure * 9/10/11 -- RAS FHI/ERI/CRI * HWATTR_KIND_0 Example: UserFlags[15:0]=smmu_msi_device_id[31:16], MasterID[15:0]=smmu_msi_device_id[15:0], ExtendedID=0 This transform might be used as part of a system-specific way of determining the DeviceID that is passed to the GIC to distinguish MSIs generated by the SMMU and those generated by client devices of the SMMU. See also: output_attribute_transform and enable_device_id_checks.
number_of_ports	int	0x1	The number of port pairs that the SMMU has.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-396 Parameters for MMU_600 (continued)

Name	Type	Default value	Description
output_attribute_transform	string	"ExtendedID[31:0]=DeviceID"	Transform the downstream attributes of a translated transaction. * "" or "none" -- the input and output attributes are identical. * How to alter the output attributes, e.g. "ExtendedID[15:0]=DeviceID[15:0], UserFlags[31]=nSSV, UserFlags[19:0]=SubstreamID" The attributes that can appear on the left hand side of the transform are ExtendedID, MasterID and UserFlags. The source attributes that can be used are: * ExtendedID/MasterID/UserFlags -- the incoming attributes. * DeviceID -- StreamID + translated_device_id_base * StreamID/SubstreamID/SSV/SEC_SID * nSEC_SID/nSSV -- the negative logic versions. * St1PBHA/St2PBHA -- the Page Based Hardware Attributes from any used leaf descriptors (or zero if not used). * STE_IMPDEF1 -- STE[127:116] The right hand side may also contain numeric literals. Any bits of the attributes that have no transform specified are retained from the input. The StreamID has had ns_sid_high/s_sid_high ORred into it for the appropriate TBU.
prefetch_only_requests	int	0x0	The simulator supports 'prefetch-only' DMI requests, which can occur for anytime for any reason and are intended to be invisible to the end execution of the model and to the user. 0 -- deny all prefetch-only requests 1 -- use debug requests for any page table walks -- form and use debug TLB/cache entries -- any faults will not record, but deny the prefetch request 2 -- treat prefetch-only requests like normal transactions -- use normal page table walk transactions -- use and form normal TLB/cache entries -- faults will alter the programmer visible state of the SMMU 0 is the safest. 1 treats the access like a debug request and requires that debug page table walks are treated correctly downstream. Any descriptors that need HTTU to allow the transaction to proceed will fail the request. 2 is dangerous, it use real transactions and reports faults that are unphysical. Real transactions can be wait()ed and this disobeys the SystemC spec for get_direct_mem_ptr().
sec_override	bool	0x0	The IMP DEF port sec_override controls whether some of the registers are accessible to secure or non-secure transactions. This parameter is the default value assumed for that port if the port is not driven by a signal.
seed	int	0x12345678	Used to seed the pseudo-random number generator that the SMMU model uses.
size_of_cd_cache	int	0x0	The number of entries in the cache holding CD structures. If this is zero then it is treated as a large number ('infinite') but it is bounded so that the host memory usage of the cache is also bounded.
size_of_l1cd_cache	int	0x0	The number of entries in the cache holding L1CD descriptors. If this is zero then it is treated as a large number ('infinite') but it is bounded so that the host memory usage of the cache is also bounded.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-396 Parameters for MMU_600 (continued)

Name	Type	Default value	Description
size_of_l1ste_cache	int	0x0	The number of entries in the cache holding L1STE descriptors. If this is zero then it is treated as a large number ('infinite') but it is bounded so that the host memory usage of the cache is also bounded.
size_of_ste_cache	int	0x0	The number of entries in the cache holding STE structures. If this is zero then it is treated as a large number ('infinite') but it is bounded so that the host memory usage of the cache is also bounded.
size_of_tlb	int	0x0	The number of entries in the TLB. If this is zero then it is treated as a large number ('infinite') but it is bounded so that the host memory usage of the cache is also bounded.
smmu_msi_device_id	int	0x0	When appropriately enabled, assume that MSIs that are generated by the SMMU are presented to the GIC with this DeviceID. See parameter msi_attribute_transform and enable_device_id_checks.
sup_cohacc	bool	0x1	The value to put in SMMU_IDR0.COHACC
sup_oas	int	0x5	The hardware has an input port sup_oas[2:0] that indicates what output address size (OAS) the _system_ has. This is sampled at reset. The model does not have this port as it is expected to be a constant for the system and not to change. Instead it is just a parameter. The allowed values are: 0 -- 32 bits 1 -- 36 bits 2 -- 40 bits 3 -- 42 bits 4 -- 44 bits 5 -- 48 bits
tlb_when_do_f_tlb_conflict_on_overlap	int	0x0	If a TLB entry is created by a walk and it overlaps an existing entry. Then there are some architectural situations where the result is known. For all others, then an implementation is allowed to use an UNPREDICTABLE combination of the two entries, or it can generate F_TLB_CONFLICT: 0 -- never generate 1 -- sometimes generate 2 -- always generate Conflicts between global and non-global entries are not detected by the model.
translated_device_id_base	int	0x0	When appropriately enabled, assume that client device accesses are translated to a DeviceID as seen by the GIC of: StreamID + translated_device_id_base See parameter output_attribute_transform and enable_device_id_checks.
tw_qs_attribute_transform	string	""	This transforms the attributes of table walk and queue transactions. * ""/"none" -- no transform * How to alter the output attributes. An example is: "ExtendedID[35:32]=HWATTR_KIND_0 The attributes that can appear on the left hand side of the transform are ExtendedID, MasterID and UserFlags. The source attributes that can be used are: * ExtendedID/MasterID/UserFlags -- the incoming attributes. * HWATTR_KIND_0 -- PBHA information The right hand side may also contain numeric literals. Any bits of the attributes with no transform are unchanged. NOTE that MPAM information comes from the parameter mpam_attribute_transform. See also output_attribute_transform and msi_attribute_transform.
version	string	"r0p0"	The version of this product

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-396 Parameters for MMU_600 (continued)

Name	Type	Default value	Description
wait_cmdq_ticks	int	0x0	This is the time to wait before doing something on the command queue. If bit 32 is set (0x1_0000_0000) then the time waited for is a uniform randomly distributed time [0,max(2,(t & 0xFFFFffff))).
wait_eventq_ticks	int	0x0	This is the time to wait before doing something on the event queue. If bit 32 is set (0x1_0000_0000) then the time waited for is a uniform randomly distributed time [0,max(2,(t & 0xFFFFffff))).
wait_imp_def_work_ticks	int	0x0	This is the time to wait before doing an IMP DEF operation. If bit 32 is set (0x1_0000_0000) then the time waited for is a uniform randomly distributed time [0,max(2,(t & 0xFFFFffff))). The IMP DEF work in this case is the number of ticks between raising pmusnapshot_req and pmusnapshot_ack being raised, and the converse operation.
wait_msi_ticks	int	0x0	This is the time to wait before sending an MSI. If bit 32 is set (0x1_0000_0000) then the time waited for is a uniform randomly distributed time [0,max(2,(t & 0xFFFFffff))).
wait_pri_req_ticks	int	0x0	This is the time to wait before processing a PRI Request. If bit 32 is set (0x1_0000_0000) then the time waited for is a uniform randomly distributed time [0,max(2,(t & 0xFFFFffff))).
wait_pri_resp_ticks	int	0x1	This is the time to wait before sending a PRI Response back to the PCIe subsystem. When a PRI Response is an auto-response then the ATC might immediately make a new ATS request, that immediately fails, that immediately makes a PRI Request, that auto-responds, etc. To break this loop, then we introduce a minimum time on all PRI Responses to give other components in the system a chance to run. If bit 32 is set (0x1_0000_0000) then the time waited for is a uniform randomly distributed time [0,max(2,(t & 0xFFFFffff))).

3.9.39 MMU_700

This model is written in C++.

MMU_700 contains the following CADI targets:

- MMU_700

MMU_700 contains the following MTI components:

- *4.85 MMU_700 - trace* on page 4-2664
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760

About MMU_600 and MMU_700

MMU_600 and MMU_700 are SMMUv3-compliant devices and are used for I/O virtualization of devices.

The hardware is a distributed SMMU. It consists of the following:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

- A single Translation Control Unit (TCU), which has a port for the programming interface of the SMMU, receives DVM messages, and does all the page walking and queue manipulation, for example.
- One or more Translation Bus Units (TBUs), which translate transactions from upstream client devices into downstream transactions.
- Zero or more connections to PCIe Root Complexes (PCIe-RCs). A maximum of 62 TBUs and PCIe-RCs can be attached.
- An interconnect that connects the TBUs and PCIe-RCs to the TCU.

A PCIe-RC has one connection to the TCU to make ATS requests but the PCIe-RC uses one or more TBUs to transform the transactions and pass them to the memory system. In the model, the TBUs used are listed in the parameter `list_of_pcie_mode`. The SMMU does not know which TBUs a particular PCIe-RC is attached to.

The TBU ORs a value into each StreamID that it receives. In the model, this is configured by the following parameters:

- `list_of_s_sid_high_at_bitpos0`
- `list_of_ns_sid_high_at_bitpos0`

The TCU, TBU, and the interconnect are all represented by this single model component.

In the model, the `tbs_pvbus_s[i]` and `tbm_pvbus_m[i]` port pair represent a TBU `i`, or `tbs_pvbus_s[i]` represents an incoming connection for a PCIe-RC. The corresponding reverse connection from the TCU to the PCIe-RC is by a special bus called `pvbus_id_routed_m` that is used to transport ATC Invalidates to the PCIe-RC.

To reduce system construction complexity, the `tbs_pvbus_s[i]` and `tbm_pvbus_m[i]` pair also acts as a TBU so that the PCIe-RC does not need to separate its normal transactions and its ATS requests.

However, ATC Invalidates are only sent to a port which appears in `list_of_pcie_rc`. It should be uniquely decoded to a single port based on `list_of_ns_sid_high_at_bitpos0`, and ATC Invalidates must be routed to the correct PCIe subsystem to invalidate the cache of ATS Responses in the subsystem. Therefore all TBUs that a PCIe-RC uses must have a unique reverse mapping from stream id to port.

Note

A bad configuration renders the model inactive.

Some configuration can be adjusted by configuration pins. These are only sampled at the negative edge of the reset pin. If you want to use these pins, then you must drive them before sending a negative edge on the reset pin. During simulation_reset, the component driving them must also drive this transition again.

The pin `sup_oas` is not supported, instead it is a parameter, as it is assumed that it would be tied to a fixed value in any specific platform.

Debug reads to the registers do not disturb the state.

Writes to registers with Update flags, including debug writes, are ignored if the Update flag is already set to one.

Debug and real accesses to the registers must be 32 bits or 64 bits.

MSIs are issued on the `qtw_pvbus_m` port using attributes that are determined by the parameter `msi_attribute_transform`, while Event queue writes are always issued with `ExtendedID=0, UserFlags=0, MasterID=0xFFFFffff`. In the hardware, there is no way of distinguishing Event queue writes from MSI writes. However, this provides a mechanism for the model system to distinguish them.

The hardware only has a single cacheability attribute for input transactions, but `PVBus` transports have both inner and outer cacheability.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

For non-PCIe-mode TBUs, whose index does not appear in `list_of_pcie_mode` or `list_of_pcie_rc`, for non-cache maintenance operations:

- If the input attribute is any type of device then it is well defined as being outer-shared and Device-nGnRnE or Device-nGnRE. There is no support for Gathering or Reordering.
- If the outer cacheable input attribute is normal, and it is Write-back, this is converted to Inner Write-back Outer Write-back (iWB-oWB) with the desired shareability. No Transient hint is supported and is always treated as non-transient.
- This leaves all other normal memory types that are mapped to Inner Normal Non-cacheable, Outer Normal Non-cacheable outer shared (iNC-oNC-osh).

Therefore, the upstream devices must present the cacheability in the *outer* cacheability attribute on `PVBus` if it is cacheable. If it is a device type then both the inner and outer attributes must be set to the same value. If it is iNC-oNC-osh, then it must be presented as such.

For PCIe-mode TBUs, that is, whose index appears in `list_of_pcie_mode` or `list_of_pcie_rc`:

- Input transactions are from PCIe and the only indication of the memory type is in the NoSnoop bit of the transaction. No shareability is transported. NoSnoop is interpreted as iNC-oNC-osh. ! NoSnoop is interpreted as iWB-oWB-ish (note Inner Shareable).
- If a NoSnoop transaction has an attribute transform applied to it and the result of the transform is weaker than iNC-oNC-osh, it is forced to iNC-oNC-osh. For example, if a NoSnoop transaction uses a page table and is transformed to iWB-oWB-nsh then it is forced to iNC-oNC-osh. However, if the page tables transformed it to a device type, then as all device types are stronger than iNC-oNC-osh, it exits the SMMU as the device type.
- In the model, transactions are classified by their incoming memory attributes as to whether they are NoSnoop or not and then are normalized appropriately:
  - iWB-oWB-any-shareability are interpreted as ! NoSnoop, therefore are normalized to iWB-oWB-ish.
  - Anything else is considered NoSnoop, and therefore is normalized to iNC-oNC-osh.
- Translated accesses also have the same interpretation to determine NoSnoop and how they are normalized. Therefore they could enter the system with different attributes to if they entered the SMMU as Untranslated Accesses and were translated by the normal translation process, before exiting downstream of the SMMU and entering the system
- It is expected that there is a component downstream of the SMMU that is aware of the system address map and will override the memory type to `device` for any transaction that accesses a peripheral.

The hardware has a single cacheability on input, and, for transactions that are neither cache-maintenance operations nor PCIe transactions, it normalizes the input to an architectural form before performing the SMMUv3 architectural transform:

- Any device type is left untouched. The input can only represent Device-nGnRE and Device-nGnRnE.
- If the input is Write-back (WB), it is normalized to iWB-oWB with the incoming shareability.
- If the input is anything else, it is normalized to iNC-oNC-osh.

The model accepts full architectural attributes of two levels of cacheability and so has to decide how to interpret this in terms of the hardware. For transactions that are not cache maintenance operations, the model replicates the outer attribute into the inner attribute and then performs the normalization that the hardware does.

The hardware normalizes the architectural output attributes and outputs a single level of cacheability and a user flag (OC) specifying if the architectural attributes were cacheable in the outer cacheable domain. If the transaction is classified as a PCIe transaction, the NoSnoop transform previously described is applied. That is, if the original transaction was NoSnoop then any weaker memory type is strengthened to iNC-oNC-osh, and the following transform is applied:

```
if      iWB-oWB-nsh/ish/osh     then output WB-nsh/ish/osh, OC = 1
else if i(NC/WT/WB)-o(WB/WT)    then output NC-Sys,         OC = 1
else if i(NC/WT/WB)-oNC         then output NC-Sys,         OC = 0
else if Device-(GRE/nGRE/nGnRE) then output DV-Sys,         OC = 0
else                                 output SO-Sys,         OC = 0
```


Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

The model only normalizes according to PCIe but otherwise leaves the architectural attributes intact on the output bus.

Important

The model deals with groups of transactions with the same attributes and a similar range of addresses. The mapping that is used is stored by the bus infrastructure and is used for subsequent sufficiently similar transactions without needing the intervention of the SMMU model, and therefore are not traced.

Model limitations

- The Performance Monitoring Unit (PMU) has limited functionality. It is intended for demonstration purposes only and does not implement all architecturally required events. It does not implement the `pmusnapshot_ack` or `pmusnapshot_req` interface.
- The model does not implement:
  - RAS.
  - Power control.
  - `sup_oas`, which controls the OAS of the SMMU. This is expected to be constant for a system.
  - The SYSCO interface.
  - The Low-Power Interface
- Cache maintenance operations cannot be inserted into the TBU ports of the SMMU.
- `PVBus` has no representation of the cache stash operations, so they are not supported.
- `TCU_CFG.XLATE_SLOTS` is fixed at 512.
- `TCU_STATUS.GNT_XLATE_SLOTS` always reads as 512.
- The `HWATTR` side-band signal is available by configuring `output_attribute_transform`, `msi_attribute_transform`, and `tw_qs_attribute_transform` with `HWATTR_KIND_0`. For some transactions, the `HWATTR` comes from the `SMMU_s_AGBPA[3:0]`. In the hardware, this register has an Update bit[31] that should be written as 1 and is turned to zero when all transactions using the old value have completed. The model does not implement this behavior and the Update bit is RAZ/WI.

Table 3-397 Ports

Name	Protocol	Type	Description
`clk_in`	`ClockSignal`	Slave	Clock signal (in RTL aclk) This is a clock time-base used by the TCU to spread some of its processing over time, if enabled by the wait_* parameters. The clock must always be connected.
`cmd_sync_irpt_ns`	*Signal* on page 2-85	Master	Pulsed interrupt output signal for non-secure CMD_SYNC having a completion signal of SIG_IRQ.
`cmd_sync_irpt_s`	*Signal* on page 2-85	Master	Pulsed interrupt output signal for secure CMD_SYNC having a completion signal of SIG_IRQ.
`event_q_irpt_ns`	*Signal* on page 2-85	Master	Pulsed interrupt output signal for the non-secure event queue becoming non-empty.
`event_q_irpt_s`	*Signal* on page 2-85	Master	Pulsed interrupt output signal for the secure event queue becoming non-empty.
`evento`	*Signal* on page 2-85	Master	Event signal
`global_irpt_ns`	*Signal* on page 2-85	Master	Pulsed interrupt output signal for non-secure SMMU_GERROR(N) signalling an error.
`global_irpt_s`	*Signal* on page 2-85	Master	Pulsed interrupt output signal for secure SMMU_S_GERROR(N) signalling an error.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-397 Ports (continued)

Name	Protocol	Type	Description
identify	*SMMUv3AEMIdentifyProtocol* on page 2-78	Master	Map the transaction to the tuple (StreamID, SubStreamID, SubStreamIDValid, SSD) The StreamID that is produced by the implementation of this protocol is not the final StreamID. The final StreamID is produced by using the list_of_ns_sid_high_at_bitpos0/ list_of_s_sid_high_at_bitpos0 parameter to map the StreamID based on the upstream port index. Also see the parameter howto_identify which can replace the functionality of this port under certain circumstances.
pri_q_irpt_ns	*Signal* on page 2-85	Master	Pulsed interrupt output signal for the PRI queue becoming non-empty.
prog_pvbus_s	PVBus	Slave	Register slave port (in RTL PROG)
pvbus_id_routed_m[62]	PVBus	Master	This is a special "id-routed" port for transmitting ATC invalidates upstream into the PCIe EndPoints, it is not a normal bus. The FastModels ATC Invalidate and PRI Response protocol specifies how to route and deal with this port.
qtw_pvbus_m	PVBus	Master	Master port used for Table Walks, MSIs and Queue access when separate_tw_msi_qs_port==true (in RTL QTW)
sec_override	*Signal* on page 2-85	Slave	Allow certain registers to be accessible to non-secure accesses from reset, as described in the TCU_SCR register.
sup_btm	*Signal* on page 2-85	Slave	System supports BTM and will be reflected in the IDR registers. This signal can override the value set by the parameters configuring the IDR registers. If BTM (Broadcast Table Maintenance) is not supported then DVM messages will be ignored.
sup_cohacc	*Signal* on page 2-85	Slave	System supports COHACC and will be reflected in the IDR registers.
sup_sev	*Signal* on page 2-85	Slave	System supports SEV and will be reflected in the IDR registers. This signal can override the value set by the parameters configuring the IDR registers.
tbm_pvbus_m[62]	PVBus	Master	The TBU master ports that carry transactions that have been translated from the correspondingly numbered tbs_pvbus_s[] port.
tbs_pvbus_s[62]	PVBus	Slave	The TBU slave ports that receive transactions to be translated. They will exit the SMMU through the same numbered pvbus_m[] port.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-397 Ports (continued)

Name	Protocol	Type	Description
tbu_cri_irpt[62]	*Signal* on page 2-85	Master	Critical error interrupt for RAS events from the TBU. This is used for specific uncorrected errors where a System Coprocessor (SCP) might have to reset the system because the stability of the system can no longer be guaranteed.
tbu_eri_irpt[62]	*Signal* on page 2-85	Master	Error Recovery Interrupt for RAS events from the TBU. This is used for uncorrected errors.
tbu_fhi_irpt[62]	*Signal* on page 2-85	Master	Fault Handling Interrupt for RAS events from the TBU. Usually this is for corrected errors.
tbu_pmu_irpt[62]	*Signal* on page 2-85	Master	TBU Performance Monitoring Unit interrupt, one per TBU.
tbu_pmusnapshot_ack[62]	*Signal* on page 2-85	Master	PMU snapshot interface for the TBU, ack a snapshot.
tbu_pmusnapshot_req[62]	*Signal* on page 2-85	Slave	PMU snapshot interface for the TBU, request a snapshot.
tbu_reset_in[62]	*Signal* on page 2-85	Slave	Reset signals The TBUs can have independent reset signals. Each signal tbu_reset_in[n] corresponds to the TBU using tbs_pvbus_s[n]/ tbs_pvbus_s[n] pair. If the SMMU receives a transaction whilst the TBU is expected to be in reset then it will complain using the ArchMsg.Warning.warning trace source. Those tbu_reset_in that correspond to a PCIe-RC connection can be connected to monitor the PCIe-RC's reset signal. If it receives an ATS request when in reset then it will complain in a similar way. You must connect these pins if you wish the TCU_NODE_STATUS for the nodes to be accurate (including any connected to the PCIe-RC).
tcu_cri_irpt	*Signal* on page 2-85	Master	Critical error interrupt for RAS events from the TCU. This is used for specific uncorrected errors where a System Coprocessor (SCP) might have to reset the system because the stability of the system can no longer be guaranteed.
tcu_eri_irpt	*Signal* on page 2-85	Master	Error Recovery Interrupt for RAS events from the TCU. This is used for uncorrected errors.
tcu_fhi_irpt	*Signal* on page 2-85	Master	Fault Handling Interrupt for RAS events from the TCU. Usually this is for corrected errors.
tcu_pmu_irpt	*Signal* on page 2-85	Master	TCU Performance Monitoring Unit interrupt
tcu_pmusnapshot_ack	*Signal* on page 2-85	Master	PMU snapshot interface for the TCU, ack a snapshot.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-397 Ports (continued)

Name	Protocol	Type	Description
tcu_pmusnapshot_req	*Signal* on page 2-85	Slave	PMU snapshot interface This is a four-phase handshake to have the corresponding PMCG perform a capture of the current counter values. PMU snapshot interface for the TCU, request a snapshot.
tcu_reset_in	*Signal* on page 2-85	Slave	The reset signal to the TCU interface.

Table 3-398 Parameters for MMU_700

Name	Type	Default value	Description
all_error_messages_through_trace	bool	0x0	Some conditions in the SMMU are so strange that the software programming the SMMU has done something wrong. At this point messages are output to either ArchMsg.Error.* or ArchMsg.Warning.* or to the error stream of the simulator. Outputting to the error stream of the simulator may cause it to return with a non-zero exit status. If you set this option to true then instead of using the error stream of the simulator it will always use a trace stream allowing the simulation to exit with a zero exit status.
behaviour_of_sampled_at_reset_signals	int	0x0	Some configuration signals into the SMMU are sampled on negedge of reset. However, it can sometimes be hard to arrange to drive a configuration pin before the negedge of reset. The configuration pins are sampled: 0 -- at negedge reset. 1 -- at negedge reset, but if a later change occurs at the same simulated time, and no transactions have occurred, then they will be resampled and the SMMU reset again.
cmdq_max_number_of_commands_to_buffer	int	0xa	The command queues can buffer fetched commands before issuing them. This parameter is roughly the maximum number of commands to do this for. The programmer visible effects are that just because the CONS pointer shows a command has been _consumed_ does not necessarily mean that it has been issued (and completed). Higher values will accentuate this effect.
enable_device_id_checks	bool	0x1	If this parameter is true then the DeviceIDs seen by the GIC are: * for client devices: DeviceID = StreamID + translated_device_id_base * for SMMU-generated MSIs: smmu_msi_device_id This parameter enables two checks: * If the DeviceID is used in the output_attribute_transform then if it overflows 32 bits then the model will warn. If the DeviceID is not used then it is assumed that the external agent that forms the DeviceID will warn if it overflows. * If the SMMU supports MSIs, then the model will check that the GIC will be able to distinguish an MSI generated by the SMMU from one generated by a client device. As the exact mechanism to determine the DeviceID is in the system and not necessarily under control of the SMMU then you can disable these warnings using this parameter. See also the parameters: output_attribute_transform and msi_attribute_transform.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-398 Parameters for MMU_700 (continued)

Name	Type	Default value	Description
howto_identify	string	"use-identify"	If 'use-identify' then will use the 'identify' port to determine the SSD, StreamID, SubStreamID of a transaction. Otherwise, this string extracts them from the transaction's attributes. Examples:- SEC_SID=ExtendedID[63], SSV=ExtendedID[62], SubstreamID=ExtendedID[51:32], StreamID=ExtendedID[31:0] nSEC_SID=ExtendedID[63], StreamID=ExtendedID[55:24], nSSV=ExtendedID[20], SubstreamID=ExtendedID[19:0] SEC_SID is one bit wide, true if StreamID is secure. SSV is one bit wide, true if SubstreamID is valid. The alternative symbols nSEC_SID and nSSV have the negative logic to SEC_SID and SSV. You must not use the negative and positive logic for the same symbol at the same time. However, just because you use negative logic for one symbol does not force you to use negative logic for the other. SubstreamID is 20 bits wide and StreamID is 32 bits. More complex examples: StreamID[31:24]=0, StreamID[23:0]=ExtendedID[23:0], SSV=1[0], ... Available attributes: ExtendedID, MasterID, UserFlags
list_of_ns_sid_high_at_bitpos0	string	""	A comma-separated list of values to bitwise OR into each Non-secure StreamID for each TBU/Node. Bit 0 of the value corresponds to bit 0 of the StreamID. Each TBU that is connected to a PCIe subsystem must serve a unique contiguous subset of StreamIDs as determined by their top bits. This is used in order to know which port to route ATC Invalidates to the PCIe subsystems.
list_of_pcie_mode	string	""	A comma-separated list of ranges of ports that represent TBUs that are attached to PCIe Root Complexes (PCIe-RC). A single PCIe-RC might use several TBUs and stripe accesses across them. The attributes handling for these TBUs are slightly different in that if the PCIe transaction is NoSnoop and the output attributes of the translation would be weaker than iNC-oNC-osh then the output is forced to iNC-oNC-osh. iNC-oNC-osh == "inner normal non-cacheable, out normal non-cacheable, outer shared"
list_of_pcie_rc	string	""	This is a list of ports that are connected to PCIe Root Complex (PCIe-RC) by a protocol called DTI-ATS. This port is used to transport ATS and PRI Requests to the SMMU from the PCIe-RC. In the real hardware, then the PCIe-RC uses this port for ATS/ PRI, and then the actual transactions go through separate TBUs. In the model, then this port can accept actual transactions as well. However, in the model, then the ATC Invalidates and the PRI Responses need to be transferred over the corresponding pvbus_id_routed_m port as DTI-ATS is bidirectional, but PVBus is not.
list_of_s_sid_high_at_bitpos0	string	""	A comma-separated list of values to bitwise OR into each Secure StreamID for each TBU/Node. Bit 0 of the value corresponds to bit 0 of the StreamID.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-398 Parameters for MMU_700 (continued)

Name	Type	Default value	Description
mpam_attribute_transform	string	"ExtendedID[62:55]=MPAM_PMG, ExtendedID[54:39]=MPAM_PARTID, ExtendedID[38]=MPAM_NS"	If MPAM is supported, then this is the transform applied to _all_ downstream transactions to transport the MPAM information. If MPAM is not supported then the transform is not applied, however, it is still checked that it is syntactically checked. If MPAM is supported in one security state, it is considered to be supported in all security states, even if that state has PARTID_MAX == 0 or PMG_MAX == 0 and the transform will be applied. NOTE that for translated transactions from client devices then: MPAM_NS = ! SEC_SID
msi_attribute_transform	string	"ExtendedID[31:0]=smmu_msi_device_id, MasterID=0xFFFFffff"	This parameter determines the MSI output attributes of SMMU-generated MSIs: * MasterID * UserFlags * ExtendedID of the transaction to use. This is a set of comma separated transforms on the output attributes. The right hand side is: * a numeric literal (or a slice of a literal) * the parameter smmu_msi_device_id * the symbol 'interrupt_kind' * 0/1 -- EVENTQ Secure/Non-secure * 2 -- PRIQ * 3/4 -- CMD_SYNC Secure/Non-secure * 5/6 -- GERROR Secure/Non-secure * 7/8 -- PMCG Secure/Non-Secure * 9/10/11 -- RAS FHI/ERI/CRI * HWATTR_KIND_0 Example: UserFlags[15:0]=smmu_msi_device_id[31:16], MasterID[15:0]=smmu_msi_device_id[15:0], ExtendedID=0 This transform might be used as part of a system-specific way of determining the DeviceID that is passed to the GIC to distinguish MSIs generated by the SMMU and those generated by client devices of the SMMU. See also: output_attribute_transform and enable_device_id_checks.
number_of_ports	int	0x1	The number of port pairs that the SMMU has.
output_attribute_transform	string	"ExtendedID[31:0]=DeviceID"	Transform the downstream attributes of a translated transaction. * "" or "none" -- the input and output attributes are identical. * How to alter the output attributes, e.g. "ExtendedID[15:0]=DeviceID[15:0], UserFlags[31]=nSSV, UserFlags[19:0]=SubstreamID" The attributes that can appear on the left hand side of the transform are ExtendedID, MasterID and UserFlags. The source attributes that can be used are: * ExtendedID/MasterID/UserFlags -- the incoming attributes. * DeviceID -- StreamID + translated_device_id_base * StreamID/SubstreamID/SSV/SEC_SID * nSEC_SID/nSSV -- the negative logic versions. * St1PBHA/St2PBHA -- the Page Based Hardware Attributes from any used leaf descriptors (or zero if not used). * STE_IMPDEF1 -- STE[127:116] The right hand side may also contain numeric literals. Any bits of the attributes that have no transform specified are retained from the input. The StreamID has had ns_sid_high/s_sid_high ORred into it for the appropriate TBU.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-398 Parameters for MMU_700 (continued)

Name	Type	Default value	Description
prefetch_only_requests	int	0x0	The simulator supports 'prefetch-only' DMI requests, which can occur for anytime for any reason and are intended to be invisible to the end execution of the model and to the user. 0 -- deny all prefetch-only requests 1 -- use debug requests for any page table walks -- form and use debug TLB/cache entries -- any faults will not record, but deny the prefetch request 2 -- treat prefetch-only requests like normal transactions -- use normal page table walk transactions -- use and form normal TLB/cache entries -- faults will alter the programmer visible state of the SMMU 0 is the safest. 1 treats the access like a debug request and requires that debug page table walks are treated correctly downstream. Any descriptors that need HTTU to allow the transaction to proceed will fail the request. 2 is dangerous, it use real transactions and reports faults that are unphysical. Real transactions can be wait()ed and this disobeys the SystemC spec for get_direct_mem_ptr().
sec_override	bool	0x0	The IMP DEF port sec_override controls whether some of the registers are accessible to secure or non-secure transactions. This parameter is the default value assumed for that port if the port is not driven by a signal.
seed	int	0x12345678	Used to seed the pseudo-random number generator that the SMMU model uses.
size_of_cd_cache	int	0x0	The number of entries in the cache holding CD structures. If this is zero then it is treated as a large number ('infinite') but it is bounded so that the host memory usage of the cache is also bounded.
size_of_l1cd_cache	int	0x0	The number of entries in the cache holding L1CD descriptors. If this is zero then it is treated as a large number ('infinite') but it is bounded so that the host memory usage of the cache is also bounded.
size_of_l1ste_cache	int	0x0	The number of entries in the cache holding L1STE descriptors. If this is zero then it is treated as a large number ('infinite') but it is bounded so that the host memory usage of the cache is also bounded.
size_of_ste_cache	int	0x0	The number of entries in the cache holding STE structures. If this is zero then it is treated as a large number ('infinite') but it is bounded so that the host memory usage of the cache is also bounded.
size_of_tlb	int	0x0	The number of entries in the TLB. If this is zero then it is treated as a large number ('infinite') but it is bounded so that the host memory usage of the cache is also bounded.
smmu_msi_device_id	int	0x0	When appropriately enabled, assume that MSIs that are generated by the SMMU are presented to the GIC with this DeviceID. See parameter msi_attribute_transform and enable_device_id_checks.
sup_cohacc	bool	0x1	The value to put in SMMU_IDR0.COHACC

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-398 Parameters for MMU_700 (continued)

Name	Type	Default value	Description
sup_oas	int	0x6	The hardware has an input port sup_oas[2:0] that indicates what output address size (OAS) the _system_ has. This is sampled at reset. The model does not have this port as it is expected to be a constant for the system and not to change. Instead it is just a parameter. The allowed values are: 0 -- 32 bits 1 -- 36 bits 2 -- 40 bits 3 -- 42 bits 4 -- 44 bits 5 -- 48 bits 6 -- 52 bits
tlb_when_do_f_tlb_conflict_on_overlap	int	0x0	If a TLB entry is created by a walk and it overlaps an existing entry. Then there are some architectural situations where the result is known. For all others, then an implementation is allowed to use an UNPREDICTABLE combination of the two entries, or it can generate F_TLB_CONFLICT: 0 -- never generate 1 -- sometimes generate 2 -- always generate Conflicts between global and non-global entries are not detected by the model.
translated_device_id_base	int	0x0	When appropriately enabled, assume that client device accesses are translated to a DeviceID as seen by the GIC of: StreamID + translated_device_id_base See parameter output_attribute_transform and enable_device_id_checks.
tw_qs_attribute_transform	string	""	This transforms the attributes of table walk and queue transactions. * ""/"none" -- no transform * How to alter the output attributes. An example is: "ExtendedID[35:32]=HWATTR_KIND_0 The attributes that can appear on the left hand side of the transform are ExtendedID, MasterID and UserFlags. The source attributes that can be used are: * ExtendedID/MasterID/UserFlags -- the incoming attributes. * HWATTR_KIND_0 -- PBHA information The right hand side may also contain numeric literals. Any bits of the attributes with no transform are unchanged. NOTE that MPAM information comes from the parameter mpam_attribute_transform. See also output_attribute_transform and msi_attribute_transform.
version	string	"r0p0"	The version of this product
wait_cmdq_ticks	int	0x0	This is the time to wait before doing something on the command queue. If bit 32 is set (0x1_0000_0000) then the time waited for is a uniform randomly distributed time [0,max(2,(t & 0xFFFFffff))).
wait_eventq_ticks	int	0x0	This is the time to wait before doing something on the event queue. If bit 32 is set (0x1_0000_0000) then the time waited for is a uniform randomly distributed time [0,max(2,(t & 0xFFFFffff))).
wait_msi_ticks	int	0x0	This is the time to wait before sending an MSI. If bit 32 is set (0x1_0000_0000) then the time waited for is a uniform randomly distributed time [0,max(2,(t & 0xFFFFffff))).

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-398 Parameters for MMU_700 (continued)

Name	Type	Default value	Description
wait_pri_req_ticks	int	0x0	This is the time to wait before processing a PRI Request. If bit 32 is set (0x1_0000_0000) then the time waited for is a uniform randomly distributed time [0,max(2,(t & 0xFFFFffff))).
wait_pri_resp_ticks	int	0x1	This is the time to wait before sending a PRI Response back to the PCIe subsystem. When a PRI Response is an auto-response then the ATC might immediately make a new ATS request, that immediately fails, that immediately makes a PRI Request, that auto-responds, etc. To break this loop, then we introduce a minimum time on all PRI Responses to give other components in the system a chance to run. If bit 32 is set (0x1_0000_0000) then the time waited for is a uniform randomly distributed time [0,max(2,(t & 0xFFFFffff))).

3.9.40 MemoryMappedCounterModule

Memory Mapped Counter Module for Generic Timers. This model is written in C++.

MemoryMappedCounterModule contains the following CADI targets:

- MemoryMappedCounterModule

MemoryMappedCounterModule contains the following MTI components:

- *4.91 MemoryMappedCounterModule - trace* on page 4-2732
- *4.107 PVBusSlave - trace* on page 4-2760

About MemoryMappedCounterModule

This component must be used by multicluster models. It also must be used to run a single core system with a timer that runs at a rate other than the input clock to the core.

Note

The component has two bus slave ports because the architecture specification permits you to map each set of registers at different, non-contiguous base addresses.

Table 3-399 Ports

Name	Protocol	Type	Description
clk_in	ClockSignal	Slave	This clock input determines the frequency of the Physical Count provided to the clusters connected to the cntvalueb port.
cntvalueb	*CounterInterface* on page 2-84	Master	This master port implements a private protocol between the cluster and the MemoryMappedCounterModule. This must be connected to the cntvalueb port on each cluster in the system and to the MemoryMappedCounterModule component.
counter_reset	*Signal* on page 2-85	Slave	Resets when set.
pvbus_control_s	PVBus	Slave	This slave port provides memory-mapped read write access to the control registers of the module.
pvbus_read_s	PVBus	Slave	This slave port provides memory-mapped read access to the status frame registers.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-400 Parameters for MemoryMappedCounterModule

Name	Type	Default value	Description
base_frequency	int	0x5f5e100	Reset value for CNTFID0, base frequency in Hz
cntcidr0123_C	int	0x0	Values to be returned for control-frame CIDR registers
cntcidr0123_R	int	0x0	Values to be returned for read-frame CIDR registers
cntpidr0123_C	int	0x0	Values to be returned for control-frame PIDR registers 0-3
cntpidr0123_R	int	0x0	Values to be returned for read-frame PIDR registers 0-3
cntpidr4567_C	int	0x0	Values to be returned for control-frame PIDR registers 4-7
cntpidr4567_R	int	0x0	Values to be returned for read-frame PIDR registers 4-7
diagnostics	int	0x0	Diagnostics
has_counter_scaling	bool	0x0	Implements ARMv8.4 generic counter scaling
non_arch_fixed_frequency	int	0x0	If set, ignore CNTFID0 and instead use this frequency in Hz
non_arch_start_at_default	bool	0x0	Firmware is expected to enable the timer at boot time. However, turning this parameter on is a model-specific way of enabling the counter module out of reset.
readonly_is_WI	bool	0x0	Ignore (rather than failing) on writes to read-frame
use_real_time	bool	0x0	Update the Generic Timer counter at a real-time base frequency instead of simulator time

3.9.41 MessageHandlingUnitV2

Message Handling Unit Version 2. This model is written in C++.

MessageHandlingUnitV2 contains the following CADI targets:

- MessageHandlingUnitV2

MessageHandlingUnitV2 contains the following MTI components:

- *4.92 MessageHandlingUnitV2 - trace* on page 4-2733

About MessageHandlingUnitV2

MessageHandlingUnitV2 is a unidirectional message channel with two APB interfaces, one for the sender and one for the receiver. There is one message handling unit for each sender and receiver pair.

The receiver can be powered off. A Q-Channel is provided for power control on the receiver side.

The message payload can be written directly to status registers. Multiple channels can be combined into a single message.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

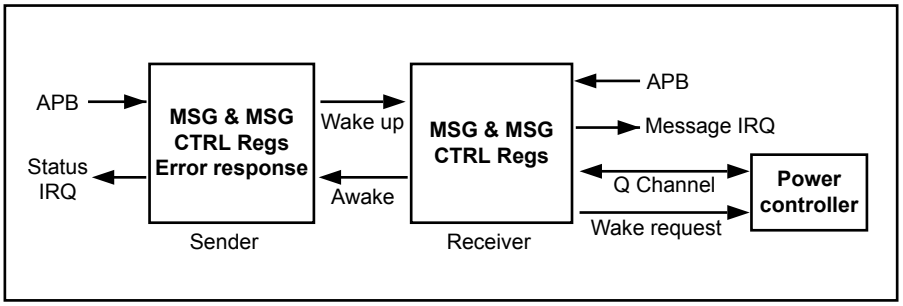

Figure 3-1 MessageHandlingUnitV2 structure

Table 3-401 Ports

Name	Protocol	Type	Description
int_access_nr2r	*Signal* on page 2-85	Master	-
int_access_r2nr	*Signal* on page 2-85	Master	-
mhu_combined_irq	*Signal* on page 2-85	Master	-
mhu_irq[124]	*Signal* on page 2-85	Master	-
mhu_snd_irq[124]	*Signal* on page 2-85	Master	-
pvbus_s_rec	PVBus	Slave	-
pvbus_s_snd	PVBus	Slave	-
qchannel_mhu_pwr	*PChannel* on page 2-80	Slave	-
reset_rec	*Signal* on page 2-85	Slave	-
reset_snd	*Signal* on page 2-85	Slave	-
snd_combined_irq	*Signal* on page 2-85	Master	-
wakerequest	*Signal* on page 2-85	Master	-

3.9.42 PL011_Uart

ARM PrimeCell UART(PL011). This model is written in LISA+.

PL011_Uart contains the following CADI targets:

- ClockDivider
- PL011_Uart

PL011_Uart contains the following MTI components:

- *4.60 ClockDivider - trace* on page 4-2438
- *4.94 PL011_Uart - trace* on page 4-2737
- *4.107 PVBusSlave - trace* on page 4-2760

About PL011_Uart

When the `untimed_fifos` parameter is false, characters of serial data are clocked to or from the SerialData port at a rate controlled by the `clk_in_ref` clock rate and the baud-rate-divider configuration of the UART clock. Enabling `untimed_fifos` permits serial data to be sent or received as fast as it can be generated or consumed. The modem control signals are still generated correctly, so the UART is not

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

able to transmit data faster than the receiving end can handle. For example, TelnetTerminal uses the CTS signal to avoid overflowing its TCP/IP buffer.

Differences between the model and the RTL

This component does not implement the DMA functionality of the PL011 PrimeCell.

Table 3-402 Ports

Name	Protocol	Type	Description
clk_in_ref	ClockSignal	Slave	Clock input, typically 14.745MHz, which sets the master transmit/receive rate.
intr	*Signal* on page 2-85	Master	Interrupt signal.
pvbus	PVBus	Slave	Slave port for register access.
serial_out	*SerialData* on page 2-77	Master	Serial input/output and control signals. Used to communicate with a serial device, such as a terminal.

Table 3-403 Parameters for PL011_Uart

Name	Type	Default value	Description
baud_rate	int	0x9600	Baud rate.
clock_rate	int	0xe10000	Clock rate for PL011.
enable_dc4	bool	0x1	Enable DC4 commands (try echo -e "help\024" in a Linux shell in a serial console)
flow_ctrl_mask_en	bool	0x0	Enable hardware flow control workaround which forcefully disables CTSen and RTSen bits in UARTCR register
in_file	string	""	Input file for data to be read by the UART
in_file_escape_sequence	string	"##"	Input file escape sequence
out_file	string	""	Output file to hold data written by the UART (use '-' to send all output to stdout)
revision	string	"r1p4"	Revision to simulate
shutdown_on_eot	bool	0x0	Shutdown simulation when a EOT (ASCII 4) char is transmitted (useful for regression tests when semihosting is not available)
shutdown_tag	string	""	Shutdown simulation when a string is transmitted
uart_enable	bool	0x0	Enable uart when the system starts up. (clock_rate and baud_rate are only valid when this option is enabled.)
unbuffered_output	bool	0x0	Unbuffered output
untimed_fifos	bool	0x1	Ignore the clock rate and transmit/receive serial data immediately

Related references
3.7.32 TelnetTerminal on page 3-636

3.9.43 PL022_SSP

ARM PrimeCell Synchronous Serial Port(PL022). This model is written in LISA+.

PL022_SSP contains the following CADI targets:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

- ClockDivider
- PL022_SSP

PL022_SSP contains the following MTI components:

- *4.60 ClockDivider - trace* on page 4-2438
- *4.107 PVBusSlave - trace* on page 4-2760

Differences between the model and the RTL

Although the PL022_SSP component has clock input, it is not internally clock-driven. This is different to the hardware.

Note

This component is a preliminary release. It is not a fully-supported peripheral.

Table 3-404 Ports

Name	Protocol	Type	Description
clk	ClockSignal	Slave	Main PrimeCell SSP clock input.
clkin	ClockSignal	Slave	PrimeCell SSP clock input.
clkout	ClockSignal	Master	Clock output.
intr	*Signal* on page 2-85	Master	Interrupt signaling.
pvbus	PVBus	Slave	Slave port for connection to PV bus master/decoder.
rorintr	*Signal* on page 2-85	Master	Receive overrun interrupt.
rtintr	*Signal* on page 2-85	Master	Receive timeout interrupt. We don't implement time out interrupt.
rx_dma_port	*PL080_DMAC_DmaPortProtocol* on page 2-74	Master	PrimeCell SSP receive DMA port.
rxd	*Value* on page 2-85	Slave	PrimeCell SSP receive data.
rxintr	*Signal* on page 2-85	Master	Receive FIFO service request port.
tx_dma_port	*PL080_DMAC_DmaPortProtocol* on page 2-74	Master	PrimeCell SSP transmit DMA port.
txd	*Value* on page 2-85	Master	PrimeCell SSP transmit data.
txintr	*Signal* on page 2-85	Master	Transmit FIFO service request.

3.9.44 PL030_RTC

ARM PrimeCell Real Time Clock(PL030). This model is written in LISA+.

PL030_RTC contains the following CADI targets:

- PL030_RTC

PL030_RTC contains the following MTI components:

- *4.107 PVBusSlave - trace* on page 4-2760

Table 3-405 Ports

Name	Protocol	Type	Description
clock	ClockSignal	Slave	Clock input, typically 1MHz, driving master count rate.
intr	*Signal* on page 2-85	Master	Interrupt signaling.
pvbus	PVBus	Slave	Slave port for connection to PV bus master/decoder.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

3.9.45 PL031_RTC

ARM PrimeCell Real Time Clock(PL031). This model is written in LISA+.

PL031_RTC contains the following CADI targets:

- PL031_RTC

PL031_RTC contains the following MTI components:

- *4.107 PVBusSlave - trace* on page 4-2760

About PL031_RTC

This component can provide a basic alarm function or long time base counter.

It has no impact on the performance of a PV system when idle or counting down. The component only executes code when the counter expires or during bus accesses.

Table 3-406 Ports

Name	Protocol	Type	Description
clock	ClockSignal	Slave	Clock input, typically 1MHz, driving master count rate.
intr	*Signal* on page 2-85	Master	Interrupt signaling.
pvbus	PVBus	Slave	Slave port for connection to PV bus master/decoder.

Table 3-407 Parameters for PL031_RTC

Name	Type	Default value	Description
RTCDR_reset_value	int	0x0	Reset value for RTCDR
RTCDR_use_current_time	bool	0x1	Use current Unix/POSIX time for reset value for RTCDR. If true RTCDR_reset_value is ignored

3.9.46 PL041_AACI

ARM PrimeCell Advanced Audio CODEC Interface(PL041). This model is written in LISA+.

PL041_AACI contains the following CADI targets:

- PL041_AACI

PL041_AACI contains the following MTI components:

- *4.107 PVBusSlave - trace* on page 4-2760

About PL041_AACI

The PL041_AACI component is designed to connect to an audio output component such as AudioOutFile or AudioOut_SDL.

The ability to play audio through this component depends on the AudioOut component in use and on the performance requirements of the software running on the simulated system. The rate of FIFO draining is controlled by the audio output to which the component is connected. This might not correspond to the rate that would be expected from the reference clock.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

This component also contains a minimal register model of the LM4529 secondary codec as implemented on development boards supplied by Arm.

Note

This component is not a complete implementation of the AACI because the following functionality is not implemented:

- Audio input
- DMA access to FIFOs, rather than Programmed I/O
- Programming of the secondary codec through FIFOs rather than slot registers

Table 3-408 Ports

Name	Protocol	Type	Description
audio	*AudioControl* on page 2-71	Master	Used to communicate with an audio out device.
clk_in_ref	ClockSignal	Slave	Reference clock input, typically 25MH.
dma_rx	*PL080_DMAC_DmaPortProtocol* on page 2-74	Master	DMA receive port.
dma_tx	*PL080_DMAC_DmaPortProtocol* on page 2-74	Master	DMA transmit port.
irq	*Signal* on page 2-85	Master	Single IRQ output port.
pvbus	PVBus	Slave	Slave port for connection to PV bus master/decoder.

Table 3-409 Parameters for PL041_AACI

Name	Type	Default value	Description
enabled	bool	0x1	Host interface connection enabled

Related references

3.7.1 AudioOut_File on page 3-618

3.7.2 AudioOut_SDL on page 3-619

3.9.47 PL050_KMI

ARM PrimeCell PS2 Keyboard/Mouse Interface(PL050). This model is written in LISA+.

PL050_KMI contains the following CADI targets:

- ClockDivider
- PL050_KMI

PL050_KMI contains the following MTI components:

- *4.60 ClockDivider - trace* on page 4-2438
- *4.107 PVBusSlave - trace* on page 4-2760

About PL050_KMI

This model communicates with models of PS/2-like devices, for example a PS2Keyboard or PS2Mouse.

Table 3-410 Ports

Name	Protocol	Type	Description
clock	ClockSignal	Slave	Clock input, typically 1MHz, which sets the master transmit/receive rate.
intr	*Signal* on page 2-85	Master	Master port signaling completion of transmit or receive.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-410 Ports (continued)

Name	Protocol	Type	Description
ps2device	*PS2Data* on page 2-74	Slave	Used to communicate with a PS/2-like device.
pvbus	PVBus	Slave	Slave port for connection to PV bus master/decoder.

3.9.48 PL061_GPIO

ARM PrimeCell General Purpose Input/Output(PL061). This model is written in LISA+.

PL061_GPIO contains the following CADI targets:

- PL061_GPIO

PL061_GPIO contains the following MTI components:

- *4.107 PVBusSlave - trace* on page 4-2760

About PL061_GPIO

This component provides eight programmable inputs or outputs. Ports of different widths can be created by multiple instantiation. In addition, an interrupt interface is provided to configure any number of pins as interrupt sources.

Table 3-411 Ports

Name	Protocol	Type	Description
GPIO_In	*Value* on page 2-85	Slave	Input lines. 32-bit data in, only [7:0] is used.
GPIO_Intr	*Signal* on page 2-85	Master	Interrupt signal indicating to an interrupt controller that an interrupt occurred in one or more of the GPIO_In lines.
GPIO_MIS	*Value* on page 2-85	Master	Indicates the masked interrupt status. 32-bit data out , only [7:0] is used. NOT necessary, as the GPIOMIS can be read from address 0x418.
GPIO_Out	*Value* on page 2-85	Master	Output lines. 32-bit data out, only [7:0] is used.
pvbus	PVBus	Slave	Slave port for register access.

Table 3-412 Parameters for PL061_GPIO

Name	Type	Default value	Description
init_inputs	int	0x0	Default input values [7:0]

3.9.49 PL080_DMAC

ARM PrimeCell DMA Controller(PL080/081). This model is written in LISA+.

PL080_DMAC contains the following CADI targets:

- ClockTimerThread
- ClockTimerThread64
- PL080_DMAC
- SchedulerThread
- SchedulerThreadEvent

PL080_DMAC contains the following MTI components:

- *4.106 PVBusMaster - trace* on page 4-2759
- *4.107 PVBusSlave - trace* on page 4-2760

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

About PL080_DMAC

This component provides 8 configurable DMA channels and 16 DMA ports for handshaking with peripherals. You can configure each channel to operate in one of eight flow control modes either under DMA control or the control of the source or destination peripheral. Transfers can occur on either master channel and can optionally be endian converted on both source and destination transfers.

This component might have a significant impact on system performance in certain flow control modes. Channels configured for small bursts, or using single bursts, and with peripheral DMA handshaking could add significant overhead. The peripheral has not been fully optimized to make use of the advanced features of the PVBus model.

Table 3-413 Ports

Name	Protocol	Type	Description
clk_in	ClockSignal	Slave	Clock signal to control DMA transfer rate.
dma_port[16]	*PL080_DMAC_DmaPortProtocol* on page 2-74	Slave	request/response ports for communicating with devices.
interr	*Signal* on page 2-85	Master	DMA error interrupt signal.
intr	*Signal* on page 2-85	Master	Combined DMA error and terminal count signal.
inttc	*Signal* on page 2-85	Master	DMA terminal count signal.
pvbus0_m	PVBus	Master	Master bus interface 0 for DMA transfers.
pvbus1_m	PVBus	Master	Master bus interface 1 for DMA transfers.
pvbus_s	PVBus	Slave	Slave port for register accesses.
reset_in	*Signal* on page 2-85	Slave	System reset.

Table 3-414 Parameters for PL080_DMAC

Name	Type	Default value	Description
activate_delay	int	0x0	request delay
fifo_size	int	0x10	Channel FIFO size in bytes
generate_clear	bool	0x0	Generate clear response
max_transfer	int	0x100	Largest atomic transfer

3.9.50 PL110_CLCD

ARM PrimeCell Color LCD Controller(PL110). This model is written in LISA+.

PL110_CLCD contains the following CADI targets:

- ClockTimerThread
- ClockTimerThread64
- PL110_CLCD
- PL11x_CLCD
- SchedulerThread
- SchedulerThreadEvent

PL110_CLCD contains the following MTI components:

- *4.95 PL11x_CLCD - trace* on page 4-2738
- *4.106 PVBusMaster - trace* on page 4-2759
- *4.107 PVBusSlave - trace* on page 4-2760

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

About PL110_CLCD

This implementation provides a register model of the LCD controller.

You can connect the model through a framebuffer port to a visualization component, for example, so that LCD output can be viewed.

The implementation is optimized for situations where the majority of the framebuffer does not change. For instance, displaying full-screen video results in significantly reduced performance. Rendering pixel data into an appropriate form for the framebuffer port (rasterization) can also take a significant amount of simulation time. If the pixel data are coming from a PVBusSlave region that has been configured as memory-like, rasterization only occurs in regions where memory contents are modified.

Table 3-415 Ports

Name	Protocol	Type	Description
clk_in	ClockSignal	Slave	Master clock input, typically 24MHz, to drive pixel clock timing.
control	*Value* on page 2-85	Slave	Auxiliary control register 1.
display	*LCD* on page 2-72	Master	Connection to visualization component.
intr	*Signal* on page 2-85	Master	Interrupt signaling for flyback events.
pvbus	PVBus	Slave	Slave port for register access.
pvbus_m	PVBus	Master	DMA port for video data.

Table 3-416 Parameters for PL11x_CLCD

Name	Type	Default value	Description
pl11x_clcd.disable_snooping_dma	bool	0x0	Disable DMA snooping
pl11x_clcd.pixel_double_limit	int	0x12c	Minimum LCD pixel width before display will be zoomed
pl11x_clcd.pl11x_behavior	int	0x0	Define PL11x behaviour. 0 for PL110, 1 for PL111

Table 3-417 Parameters for PL110_CLCD

Name	Type	Default value	Description
disable_snooping_dma	bool	0x0	Disable DMA snooping
pixel_double_limit	int	0x12c	Minimum LCD pixel width before display will be zoomed

Related references

1.16 Visualisation Library on page 1-53

3.9.51 PL111_CLCD

ARM PrimeCell Color LCD Controller(PL111). This model is written in LISA+.

PL111_CLCD contains the following CADI targets:

- ClockTimerThread
- ClockTimerThread64
- PL111_CLCD
- PL11x_CLCD
- SchedulerThread
- SchedulerThreadEvent

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PL111_CLCD contains the following MTI components:

- *4.95 PL11x_CLCD - trace* on page 4-2738
- *4.106 PVBusMaster - trace* on page 4-2759
- *4.107 PVBusSlave - trace* on page 4-2760

About PL111_CLCD

This component implements the hardware cursor support of the PL111_CLCD, which is the main difference with PL110_CLCD.

Table 3-418 Ports

Name	Protocol	Type	Description
clk_in	ClockSignal	Slave	Master clock input, typically 24MHz, to drive pixel clock timing.
control	*Value* on page 2-85	Slave	Auxiliary control register 1.
display	*LCD* on page 2-72	Master	Connection to visualization component.
intr	*Signal* on page 2-85	Master	Interrupt signaling for flyback events.
pvbus	PVBus	Slave	Slave port for register access.
pvbus_m	PVBus	Master	DMA port for video data.

Table 3-419 Parameters for PL111_CLCD

Name	Type	Default value	Description
disable_snooping_dma	bool	0x0	Disable DMA snooping
pixel_double_limit	int	0x12c	Minimum LCD pixel width before display will be zoomed

Table 3-420 Parameters for PL11x_CLCD

Name	Type	Default value	Description
pl11x_clcd.disable_snooping_dma	bool	0x0	Disable DMA snooping
pl11x_clcd.pixel_double_limit	int	0x12c	Minimum LCD pixel width before display will be zoomed
pl11x_clcd.pl11x_behavior	int	0x0	Define PL11x behaviour. 0 for PL110, 1 for PL111

Related references

1.16 Visualisation Library on page 1-53

3.9.52 PL180_MCI

ARM PrimeCell Multimedia Card Interface (PL180). This model is written in LISA+.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PL180_MCI contains the following CADI targets:

- PL180_MCI

PL180_MCI contains the following MTI components:

- *4.107 PVBusSlave - trace* on page 4-2760

About PL180_MCI

When paired with an MMC card model, the PL180_MCI component provides emulation of a flexible, persistent storage mechanism. The PL180_MCI component fully models the registers of the corresponding PrimeCell, but supports a subset of the functionality of the PL180:

- The controller supports block mode transfers, but does not currently support streaming data transfer.
- The controller can be attached to a single MMC device. The MMC bus mode and SDIO modes of the PL180 PrimeCell are not supported.
- Command and Data timeouts are not simulated.
- Payload CRC errors are not simulated.
- The DMA interface present in the PL180 PrimeCell is not modeled.
- Minimal timing is implemented within the model.

Note

At compile time, you can enable command tracing within the PL180_MCI component by modifying the `PL180_TRACE` macro in the `MMC.lisa` file. This sends command and event trace to standard output. You can use this output to help diagnose device driver and controller-to-card protocol issues.

Table 3-421 Ports

Name	Protocol	Type	Description
`MCIINTR[2]`	*`Signal`* on page 2-85	Master	Interrupts.
`mmc_m`	*`MMC_Protocol`* on page 2-73	Master	The MultiMediaCard (MMC) master port.
`pvbus`	`PVBus`	Slave	Slave port for register access.

Table 3-422 Parameters for PL180_MCI

Name	Type	Default value	Description
`pl180_fifo_depth`	`int`	`0x10`	PL180 FIFO Depth

Related references

3.9.33 MMC on page 3-767

3.9.53 PL192_VIC

ARM PrimeCell Vectored Interrupt Controller(PL192). This model is written in LISA+.

PL192_VIC contains the following CADI targets:

- PL192_VIC

PL192_VIC contains the following MTI components:

- *4.107 PVBusSlave - trace* on page 4-2760

About PL192_VIC

This component aggregates interrupts and generates interrupt signals for the Arm processor. When coupled with an Arm processor that provides a VIC port, routing to the appropriate interrupt handler can optionally be performed in hardware, reducing interrupt latency. The PL192_VIC can also be daisy-

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

chained with other PL192 VICs to permit more than 32 interrupts. The VIC supports hardware and software prioritization of interrupts.

Table 3-423 Ports

Name	Protocol	Type	Description
VICIRQACK	*Signal* on page 2-85	Slave	Receive acknowledge signal from next level VIC or processor.
VICIRQACKOUT	*Signal* on page 2-85	Master	Used to send out acknowledge signals when daisy chained.
VICIntSource[32]	*Signal* on page 2-85	Slave	Interrupt source input sources.
VICVECTADDRIN	*ValueState* on page 2-85	Slave	Used to receive vector address when daisy chained.
VICVECTADDROUT	*ValueState* on page 2-85	Master	Used to send vector address to next level VIC or processor.
nVICFIQ	*Signal* on page 2-85	Master	Send out FIQ signal to the next level VIC or CPI.
nVICFIQIN	*Signal* on page 2-85	Slave	Used to receive FIQ signal when daisy chained.
nVICIRQ	*Signal* on page 2-85	Master	Send out IRQ signal to the next level VIC or procesessor.
nVICIRQIN	*Signal* on page 2-85	Slave	Used to receive IRQ signal when daisy chained.
pvbus	PVBus	Slave	Slave port for register access.

3.9.54 PL310_L2CC

ARM PrimeCell Level 2 Cache Controller (PL310). This model is written in LISA+.

PL310_L2CC contains the following CADI targets:

- PL310_L2CC

PL310_L2CC contains the following MTI components:
- *4.96 PL310_L2CC - trace* on page 4-2739

About PL310_L2CC

The presence of additional on-chip secondary cache can improve performance when significant memory traffic is generated by the processor. A secondary cache assumes the existence of a Level 1, or primary, cache that is closely coupled or internal to the processor.

This component has two modes of operation, which are controlled by the `cache-state_modelled` parameter:

Register view
Cache control registers are present but the cache behavior is not modeled.

Functional model
Cache behavior is modeled.

Arm supports the use of the PL310 when connected to the Arm Cortex-A5 or Cortex-A9 processor.

This component implements the programmer-visible functionality of the PL310, and excludes some non-programmer visible features. The following features are implemented in the model:

- Physically addressed and physically tagged.
- Lockdown format C supported, for data and instructions. Lockdown format C is also known as way locking.
- Lockdown by line supported.
- Lockdown by master ID supported.
- Direct mapped to 16-way associativity, depending on the configuration and the use of lockdown registers. The associativity is configurable as 8 or 16.
- L2 cache available size can be 16KB to 8MB, depending on configuration and the use of the lockdown registers.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

- Fixed line length of 32 bytes (8 words or 256 bits).
- Supports all of the AXI cache modes:
  - write-through and write-back.
  - read allocate, write allocate, read and write allocate.
- Force write-allocate option to always have cacheable writes allocated to L2 cache, for processors not supporting this mode.
- Normal memory non-cacheable shared reads are treated as cacheable non-allocatable. Normal memory non-cacheable shared writes are treated as cacheable write-through no write-allocate. There is an option, Shared Override, to override this behavior.
- TrustZone support, with the following features:
  - Non-Secure (NS) tag bit added in tag RAM and used for lookup in the same way as an address bit.
  - NS bit in Tag RAM used to determine security level of evictions to L3.
  - Restrictions for NS accesses for control, configuration, and maintenance registers to restrict access to secure data.
- Pseudo-Random victim selection policy. You can make this deterministic with use of lockdown registers.
- Software option to enable exclusive cache configuration.
- Configuration registers accessible using address decoding in the component.
- Interrupt triggering in case of an error response when accessing L3.
- Maintenance operations.
- Prefetching capability.

The performance of this component depends on the configuration of the associated L1 caches and the mode it is in:

- Register mode: no significant effect.
- Functional mode with functional-mode L1: the addition of a functional L2 cache has minimal further impact on performance when running applications that are cache-bound.
- Functional mode with a register-mode L1: there is a significant impact on system performance.

Differences between the model and the RTL

This model does not implement the following features, most of which are not relevant from a PV modeling point of view:

- There is no interface to the data and tag RAM as they are embedded to the model.
- Critical word first linefill is not supported, as it is not relevant for PV modeling.
- Buffers are not modeled.
- Outstanding accesses on slave and master ports cannot occur by design in a PV model as all transactions are atomic.
- Option to select one or two master ports and option to select one or two slave ports is not supported. Only one master port and one slave port is supported.
- Clock management and power modes are not supported, as they are not relevant for PV modeling.
- Wait, latency, clock enable, parity, and error support for data and tag RAMs not included, as they are not relevant for PV modeling, and the data and tag RAMs embedded in the model cannot generate error responses.
- MBIST support is not included.
- Debug mode and debug registers are not supported.
- Test mode and scan chains are not supported.
- L2 cache event monitoring is not supported.
- Address filtering in the master ports is not supported.
- Performance counters are not supported.
- Specific Cortex-A9 related optimizations are not supported: Prefetch hints, Full line of zero and Early write response.
- Hazard detection is not required because of the atomic nature of the accesses at PV modeling and the fact that buffers are not modeled, therefore hazards cannot occur.
- Registers that belong to unimplemented features are accessible but do not have any functionality.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

This model implements the following features differently to the hardware:

- Error handling. DECERR from the master port is mapped to SLVERR. Internal errors in cache RAM (like parity errors) cannot happen in the model.
- Background cache operations do not occur in the background. They occur atomically.
- The LOCKDOWN_BY_LINE and LOCKDOWN_BY_MASTER parameter values are reflected in the CacheType register, but the feature is not switched off when the parameter is 0.

This feature is additional:

- Data RAM and Tag RAM are embedded to the model.

Table 3-424 Ports

Name	Protocol	Type	Description
DECERRINTR	*Signal* on page 2-85	Master	Decode error received on master port from L3.
ECNTRINTR	*Signal* on page 2-85	Master	Event Counter Overflow / Increment.
ERRRDINTR	*Signal* on page 2-85	Master	Error on L2 data RAM read.
ERRRTINTR	*Signal* on page 2-85	Master	Error on L2 tag RAM read.
ERRWDINTR	*Signal* on page 2-85	Master	Error on L2 data RAM write.
ERRWTINTR	*Signal* on page 2-85	Master	Error on L2 tag RAM write.
L2CCINTR	*Signal* on page 2-85	Master	Combined interrupt output.
PARRDINTR	*Signal* on page 2-85	Master	Parity error on L2 data RAM read.
PARRTINTR	*Signal* on page 2-85	Master	Parity error on L2 tag RAM read.
SLVERRINTR	*Signal* on page 2-85	Master	Slave error on master port from L3.
pvbus_m	PVBus	Master	Master port for connection to PV bus master/decoder.
pvbus_s	PVBus	Slave	Slave port for connection to PV bus master/decoder.

Table 3-425 Parameters for PL310_L2CC

Name	Type	Default value	Description
ASSOCIATIVITY	int	0x0	Associativity for Auxiliary Control Register
CACHEID	int	0x0	Cache controller cache ID
CFGBIGEND	int	0x0	Big-endian mode for accessing configuration registers out of reset
LOCKDOWN_BY_LINE	int	0x0	Lockdown by line - value is reflected in CacheType register Bit 25, but the feature is not switched off when the parameter is 0
LOCKDOWN_BY_MASTER	int	0x0	Lockdown by master - value is reflected in CacheType register Bit 26, but the feature is not switched off when the parameter is 0
REGFILEBASE	int	0x1f002000	Base address for accessing configuration registers
WAYSIZE	int	0x1	Size of ways for Auxiliary Control Register
cache-state_modelled	bool	0x0	Specifies whether real cache state is modelled (vs. register model)
delay_cache_hit	int	0x0	Cost to handle a cache hit
delay_cache_miss	int	0x0	Cost to handle a cache miss
delay_cache_perbeat	int	0x0	Cost to handle one beat of cache data movement

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Related references

1.4.2 Caches in PV models on page 1-20

3.9.55 PL330_DMAC

ARM PrimeCell DMA Controller(PL330). This model is written in LISA+.

PL330_DMAC contains the following CADI targets:

- ClockTimerThread
- ClockTimerThread64
- PL330_DMAC
- SchedulerThread
- SchedulerThreadEvent

PL330_DMAC contains the following MTI components:

- *4.97 PL330_DMAC - trace* on page 4-2742
- *4.106 PVBusMaster - trace* on page 4-2759
- *4.107 PVBusSlave - trace* on page 4-2760

About PL330_DMAC

The model uses a single LISA+ component but with a C++ model for each of the channels included in the LISA+ file. Enabled channels are kept on an enabled channels stack in priority order. When a channel state changes, re-arbitration takes place to make the highest (topmost) channel active.

Each transaction carries the identity of the requesting thread. This controller has up to eight channel threads and a manager thread. Each has an ID. In the hardware, the ID is AxID[3:0], with `0x0 - (numberOfChannels - 1)` identifying channels and `(numberOfChannels)` identifying the manager. For example, `0x0-0x7` and `0x8`, respectively. The manager originates only instruction fetches, and the manager ID is also used for instruction fetches issued by the channels.

In the model, the identity of the requesting thread is encoded into each transaction using the low-order 16 bits of the Master ID field:

- Channel data: `0-7`.
- Channel instruction fetch: `0xffff`.
- Manager instruction fetch: `0xffff`.

If a downstream component needs to know the IDs of bus masters that use either the low-order 16 bits or the label, use the label. The LabellerForDMA330 component shifts the low-order 16 bits into the label, while providing a degree of control over the label encoding. The example below maintains separate IDs for each data channel while using the correct hardware ID to identify instruction fetch for a DMA-330 with 8 channels:

```
pl330_dma : PL330_DMAC( "p_max_channels" = 8 );
dma_labeller : LabellerForDMA330(
    "dma330_discriminate_data_channels" = true,
    "dma330_s_instruction_label" = 8,
    "dma330_ns_instruction_label" = 8 );
pl330_dma.pvbus_m => dma_labeller.pvbus_s;
dma_labeller.pvbus_m => output_bus.pvbus_s;
```

Table 3-426 Ports

Name	Protocol	Type	Description
clk_in	ClockSignal	Slave	Clock input.
irq_abort_master_port	*Signal on page 2-85*	Master	Undefined instruction or instruction error.
irq_master_port[32]	*Signal on page 2-85*	Master	Sets when DMASEV.
pvbus_m	PVBus	Master	Master port for all memory accesses.
pvbus_s	PVBus	Slave	Slave port for all register accesses (secure).

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-426 Ports (continued)

Name	Protocol	Type	Description
pvbus_s_ns	PVBus	Slave	Slave port for all register accesses (non-secure).
reset_in	*Signal* on page 2-85	Slave	System reset.

Table 3-427 Parameters for PL330_DMAC

Name	Type	Default value	Description
activate_delay	int	0x0	request delay
fifo_size	int	0x10	Channel FIFO size in bytes
generate_clear	bool	0x0	Generate clear response
max_transfer	int	0x100	Largest atomic transfer
p_axi_bus_width_param	int	0x20	AXI bus width
p_buffer_depth	int	0x10	buffer depth
p_cache_line_words	int	0x1	number of words in a cache line
p_cache_lines	int	0x1	number of cache lines
p_controller_boots	bool	0x1	DMA boots from reset
p_controller_nsecure	bool	0x0	Controller non-secure at reset (boot_manager_ns)
p_irq_nsecure	int	0x0	Interrupts non-secure at reset
p_lsq_read_size	int	0x4	LSQ read buffer depth
p_lsq_write_size	int	0x4	LSQ write buffer depth
p_max_channels	int	0x8	virtual channels
p_max_irqs	int	0x20	number of interrupts
p_max_periph	int	0x20	number of peripheral interfaces
p_perip_request_acceptance_0	int	0x2	Peripheral 0 request acceptance
p_perip_request_acceptance_1	int	0x2	Peripheral 1 request acceptance
p_perip_request_acceptance_10	int	0x2	Peripheral 10 request acceptance
p_perip_request_acceptance_11	int	0x2	Peripheral 11 request acceptance
p_perip_request_acceptance_12	int	0x2	Peripheral 12 request acceptance
p_perip_request_acceptance_13	int	0x2	Peripheral 13 request acceptance
p_perip_request_acceptance_14	int	0x2	Peripheral 14 request acceptance
p_perip_request_acceptance_15	int	0x2	Peripheral 15 request acceptance
p_perip_request_acceptance_16	int	0x2	Peripheral 16 request acceptance
p_perip_request_acceptance_17	int	0x2	Peripheral 17 request acceptance
p_perip_request_acceptance_18	int	0x2	Peripheral 18 request acceptance
p_perip_request_acceptance_19	int	0x2	Peripheral 19 request acceptance
p_perip_request_acceptance_2	int	0x2	Peripheral 2 request acceptance
p_perip_request_acceptance_20	int	0x2	Peripheral 20 request acceptance

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-427 Parameters for PL330_DMAC (continued)

Name	Type	Default value	Description
p_perip_request_acceptance_21	int	0x2	Peripheral 21 request acceptance
p_perip_request_acceptance_22	int	0x2	Peripheral 22 request acceptance
p_perip_request_acceptance_23	int	0x2	Peripheral 23 request acceptance
p_perip_request_acceptance_24	int	0x2	Peripheral 24 request acceptance
p_perip_request_acceptance_25	int	0x2	Peripheral 25 request acceptance
p_perip_request_acceptance_26	int	0x2	Peripheral 26 request acceptance
p_perip_request_acceptance_27	int	0x2	Peripheral 27 request acceptance
p_perip_request_acceptance_28	int	0x2	Peripheral 28 request acceptance
p_perip_request_acceptance_29	int	0x2	Peripheral 29 request acceptance
p_perip_request_acceptance_3	int	0x2	Peripheral 3 request acceptance
p_perip_request_acceptance_30	int	0x2	Peripheral 30 request acceptance
p_perip_request_acceptance_31	int	0x2	Peripheral 31 request acceptance
p_perip_request_acceptance_4	int	0x2	Peripheral 4 request acceptance
p_perip_request_acceptance_5	int	0x2	Peripheral 5 request acceptance
p_perip_request_acceptance_6	int	0x2	Peripheral 6 request acceptance
p_perip_request_acceptance_7	int	0x2	Peripheral 7 request acceptance
p_perip_request_acceptance_8	int	0x2	Peripheral 8 request acceptance
p_perip_request_acceptance_9	int	0x2	Peripheral 9 request acceptance
p_periph_nsecure	bool	0x0	Peripherals non-secure at reset
p_read_issuing_capability	int	0x1	AXI read issuing capability
p_reset_pc	int	0x60000000	DMA PC at reset
p_write_issuing_capability	int	0x1	AXI write issuing capability
revision	string	"r0p0"	revision ID

3.9.56 PL350_SMC

ARM PrimeCell Static Memory Controller(PL350). This model is written in LISA+.

PL350_SMC contains the following CADI targets:

- PL350_SMC
- PVBusExclusiveMonitor
- TZSwitch

PL350_SMC contains the following MTI components:

- *4.102 PVBusExclusiveMonitor - trace* on page 4-2752
- *4.105 PVBusMapper - trace* on page 4-2758
- *4.106 PVBusMaster - trace* on page 4-2759
- *4.107 PVBusSlave - trace* on page 4-2760

About PL350_SMC

This component provides two memory interfaces. Each interface can be connected to a maximum of four memory devices, giving a total of eight inputs from the PVBusDecoder and eight outputs to either

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SRAM or NAND devices. Only one kind of memory can be connected to a particular interface, either SRAM or NAND.

It provides a PVBus slave to control the device behavior. A remap port is also provided to assist in remapping particular memory regions.

This component is optimized to have negligible impact on transaction performance, except when memory remap settings are changed, when there might be a significant effect.

Table 3-428 Ports

Name	Protocol	Type	Description
apb_interface	PVBus	Slave	This is where we expect to receive all the APB data which is used to read/write the device regs.
axi_chip_if0_in[4]	PVBus	Slave	This is where we expect to receive all the AXI data which is used to read/wrie NAND/RAM mem.
axi_chip_if0_out[4]	PVBus	Master	Master interface 0 to connect to SRAM/NAND.
axi_chip_if1_in[4]	PVBus	Slave	This is where we expect to receive all the AXI data which is used to read/wrie NAND/RAM mem.
axi_chip_if1_out[4]	PVBus	Master	Master interface 1 to connect to SRAM/NAND.
axi_remap	PVBus	Slave	This is the remap port that the designer needs to connect to zero.
irq_in_if0	*Signal* on page 2-85	Slave	Interrupt signals from devices connected on interface 0.
irq_in_if1	*Signal* on page 2-85	Slave	Interrupt signals from device connected on interface 1.
irq_out	*Signal* on page 2-85	Master	Interrupt port.
nand_remap_port	PVBus	Slave	Remaps the connected NAND port to 0x0.

Table 3-429 Parameters for PVBusExclusiveMonitor

Name	Type	Default value	Description
apply_access_width_criteria_to_non_excl_stores	bool	0x1	Apply the given exclusive store width matching criteria to non-exclusive stores
clear_on_strex_address_mismatch	bool	0x1	Whether monitor is cleared when strex fails due to address mismatch
enable_component	bool	0x1	Enable component
log2_granule_size	int	0x0	log2 of address granule size
match_access_width	bool	0x0	Fail STREX if not the same access width as LDREX
match_secure_state	bool	0x1	Treat the secure state like an address bit
monitor_access_level	int	0x0	0: Monitor all accesses, 1: Monitor all accesses except WriteBack, 2: Only monitor accesses with memory type NonCacheable or Device
monitor_non_excl_stores	bool	0x0	Monitor non-exclusive stores from the same master

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-429 Parameters for PVBusExclusiveMonitor (continued)

Name	Type	Default value	Description
number_of_monitors	int	0x8	Number of monitors
shareability_domain	int	0x3	Maximum shareability domain of interest, transactions outside of the domain will pass through un-monitored (0-non-shared, 1-inner, 2-outer, 3-system)

Table 3-430 Parameters for TZSwitch

Name	Type	Default value	Description
normal	int	0x2	Normal Port
secure	int	0x1	Secure Port

Table 3-431 Parameters for PL350_SMC

Name	Type	Default value	Description
IF0_CHIP0	bool	0x0	Interface 0 chip 0 connected
IF0_CHIP0_BASE	int	0x0	Interface 0 chip 0 Base address
IF0_CHIP0_SIZE	int	0x0	Interface 0 chip 0 Size
IF0_CHIP1	bool	0x0	Interface 0 chip 1 connected
IF0_CHIP1_BASE	int	0x0	Interface 0 chip 1 Base address
IF0_CHIP1_SIZE	int	0x0	Interface 0 chip 1 Size
IF0_CHIP2	bool	0x0	Interface 0 chip 2 connected
IF0_CHIP2_BASE	int	0x0	Interface 0 chip 2 Base address
IF0_CHIP2_SIZE	int	0x0	Interface 0 chip 2 Size
IF0_CHIP3	bool	0x0	Interface 0 chip 3 connected
IF0_CHIP3_BASE	int	0x0	Interface 0 chip 3 Base address
IF0_CHIP3_SIZE	int	0x0	Interface 0 chip 3 Size
IF0_MEM_TYPE_PARAMETER	int	0x0	Interface 0 Mem type
IF1_CHIP0	bool	0x0	Interface 1 chip 0 connected
IF1_CHIP0_BASE	int	0x0	Interface 1 chip 0 Base address
IF1_CHIP0_SIZE	int	0x0	Interface 1 chip 0 Size
IF1_CHIP1	bool	0x0	Interface 1 chip 1 connected
IF1_CHIP1_BASE	int	0x0	Interface 1 chip 1 Base address
IF1_CHIP1_SIZE	int	0x0	Interface 1 chip 1 Size
IF1_CHIP2	bool	0x0	Interface 1 chip 2 connected
IF1_CHIP2_BASE	int	0x0	Interface 1 chip 2 Base address
IF1_CHIP2_SIZE	int	0x0	Interface 1 chip 2 Size

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-431 Parameters for PL350_SMC (continued)

Name	Type	Default value	Description
IF1_CHIP3	bool	0x0	Interface 1 chip 3 connected
IF1_CHIP3_BASE	int	0x0	Interface 1 chip 3 Base address
IF1_CHIP3_SIZE	int	0x0	Interface 1 chip 3 Size
IF1_MEM_TYPE_PARAMETER	int	0x0	Interface 1 Mem type
PERIPH_ID_0	int	0x52	Periph_ID_0 value
REMAP	int	-0x1	Remap the device
revision	string	"r1p2"	Revision

3.9.57 PL350_SMC_NAND_FLASH

A NAND Flash implementation which works with PL350. This model is written in LISA+.

PL350_SMC_NAND_FLASH contains the following CADI targets:

- PL350_SMC_NAND_FLASH

PL350_SMC_NAND_FLASH contains the following MTI components:

- *4.107 PVBusSlave - trace* on page 4-2760

About PL350_SMC_NAND_FLASH

Program the component as you would the hardware.

Table 3-432 Ports

Name	Protocol	Type	Description
irq	*Signal* on page 2-85	Master	Interrupt signaling.
pvbus	PVBus	Slave	Slave port for connection to PV bus master/decoder.

Table 3-433 Parameters for PL350_SMC_NAND_FLASH

Name	Type	Default value	Description
DEVICE_1	int	0xec	Device manufacturer code
DEVICE_2	int	0xda	Device code
DEVICE_3	int	0x80	Device 3rd cycle code
DEVICE_4	int	0x15	Device 4th cycle code
DEVICE_NAME	string	"Samsung K9F1G08U0M"	Device Name
NAND_BLOCK_COUNT	int	0x800	number of blocks in the flash device
NAND_FLASH_SIZE	int	0x10800000	flash size in byte
NAND_PAGE_COUNT_PER_BLOCK	int	0x40	number of pages in each block
NAND_PAGE_SIZE	int	0x840	page size
NAND_SPARE_SIZE_PER_PAGE	int	0x40	Spare size per page
NAND_VALID_SIZE_PER_PAGE	int	0x800	valid page size

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

3.9.58 PL370_HDLCD

ARM PrimeCell HD Color LCD Controller (Nominal Designation PL370). This model is written in LISA+.

PL370_HDLCD contains the following CADI targets:

- ClockTimerThread
- ClockTimerThread64
- PL370_HDLCD
- SchedulerThread
- SchedulerThreadEvent

PL370_HDLCD contains the following MTI components:

- *4.106 PVBusMaster - trace* on page 4-2759
- *4.107 PVBusSlave - trace* on page 4-2760

Note

Too fast a pixel clock can slow the rest of the simulation.

Table 3-434 Ports

Name	Protocol	Type	Description
clk_in	ClockSignal	Slave	Master clock input, typically 24MHz, to drive pixel clock timing.
display	*LCD on page 2-72*	Master	Connection to visualization component.
intr	*Signal on page 2-85*	Master	Interrupt signaling line for flyback events.
pvbus	PVBus	Slave	Slave port for connection to PV bus master/decoder.
pvbus_m	PVBus	Master	DMA port for collecting video data from memory/framebuffer.

Table 3-435 Parameters for PL370_HDLCD

Name	Type	Default value	Description
diagnostics	int	0x0	Diagnostics level
disable_snooping_dma	bool	0x0	Disable DMA snooping
force_frame_rate	int	0x32	Force frame rate to the value of the parameter in frames per simulated second, regardless of the input clock. When 0, use the input clock as a pixel clock

3.9.59 PL390_GIC

Generic Interrupt Controller (PL390). This model is written in LISA+.

PL390_GIC contains the following CADI targets:

- PL390_GIC

PL390_GIC contains the following MTI components:

- *4.107 PVBusSlave - trace* on page 4-2760

About PL390_GIC

The GIC provides support for three interrupt types:

- Software Generated Interrupts (SGI)
- Private Peripheral Interrupts (PPI)
- Shared Peripheral Interrupts (SPI)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

You can set:

- Security state for an interrupt
- Priority state for an interrupt
- Enabling or disabling state for an interrupt
- Processors that receive an interrupt

A processor interface consists of a pair of interfaces called `pvbus_cpu` and `pvbus_distributor`. The `enable_c<n>` and `match_c<n>` signals identify the originator of a transaction on `pvbus_cpu`. Similarly, `enable_d<n>` and `match_d<n>` signals identify the originator of a transaction on `pvbus_distributor`. <n> corresponds to the number of a processor interface.

Note

To reduce compile time, the registers are not available by default. To activate them, uncomment either of the following statements in `PL390_GIC.lisa`:

```
// #define FEW_CADI_REGISTER
// #define ALL_CADI_REGISTER
```

Table 3-436 Ports

Name	Protocol	Type	Description
cfgsdisable	*Signal* on page 2-85	Slave	Set preventing write accesses to security-critical configuration registers.
enable_c[8]	*ValueState* on page 2-85	Slave	Compared with masked PVBus master id to select processor interface: (master_id & enable_c<n>) == match_c<n>.
enable_d[8]	*ValueState* on page 2-85	Slave	Compared with masked PVBus master id to select distributor interface: (master_id & enable_d<n>) == match_d<n>.
legacy_nfiq[8]	*Signal* on page 2-85	Slave	Legacy FIQ interrupt for processor Interface <n>.
legacy_nirq[8]	*Signal* on page 2-85	Slave	Input interrupt signals.
match_c[8]	*ValueState* on page 2-85	Slave	Mask on the PVBus master id to select processor interface: (master_id & enable_c<n>) == match_c<n>.
match_d[8]	*ValueState* on page 2-85	Slave	Mask on the PVBus master id to select distributor interface: (master_id & enable_d<n>) == match_d<n>.
nfiq[8]	*Signal* on page 2-85	Master	Send out FIQ signal to processor <n>.
nirq[8]	*Signal* on page 2-85	Master	Send out IRQ signal to processor <n>.
ppi_c0[16]	*Signal* on page 2-85	Slave	Private peripheral interrupt for processor 0 (num_cpus> = 1).
ppi_c1[16]	*Signal* on page 2-85	Slave	Private peripheral interrupt for processor 1 (num_cpus> = 2).
ppi_c2[16]	*Signal* on page 2-85	Slave	Private peripheral interrupt for processor 2 (num_cpus> = 3).
ppi_c3[16]	*Signal* on page 2-85	Slave	Private peripheral interrupt for processor 3 (num_cpus> = 4).
ppi_c4[16]	*Signal* on page 2-85	Slave	Private peripheral interrupt for processor 4 (num_cpus> = 5).
ppi_c5[16]	*Signal* on page 2-85	Slave	Private peripheral interrupt for processor 5 (num_cpus> = 6).
ppi_c6[16]	*Signal* on page 2-85	Slave	Private peripheral interrupt for processor 6 (num_cpus> = 7).
ppi_c7[16]	*Signal* on page 2-85	Slave	Private peripheral interrupt for processor 7 (num_cpus> = 8).
pvbus_cpu	PVBus	Slave	Slave port for connection to processor interface.
pvbus_distributor	PVBus	Slave	Slave port for connection to distributor interface.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-436 Ports (continued)

Name	Protocol	Type	Description
reset_in	*Signal* on page 2-85	Slave	Reset signal.
spi[988]	*Signal* on page 2-85	Slave	Shared peripheral interrupt inputs.

Table 3-437 Parameters for PL390_GIC

Name	Type	Default value	Description
ARCHITECTURE_VERSION	int	0x1	set architecture version in periph_id register
AXI_IF	bool	0x1	set interface type in peripheral identification register 8
C_ID_WIDTH	int	0x20	width of the cpu interface master id
D_ID_WIDTH	int	0x20	width of the distributor interface master id
ENABLE_LEGACY_FIQ	bool	0x1	provide legacy fiq interrupt inputs
ENABLE_LEGACY_IRQ	bool	0x1	provide legacy irq interrupt inputs
ENABLE_PPI_EDGE	bool	0x0	ppi edge sensitive
ENABLE_TRUSTZONE	bool	0x1	support trustzone
INIT_ENABLE_C0	int	0xffffffff	initial value of register ENABLE_C0
INIT_ENABLE_C1	int	0xffffffff	initial value of register ENABLE_C1
INIT_ENABLE_C2	int	0xffffffff	initial value of register ENABLE_C2
INIT_ENABLE_C3	int	0xffffffff	initial value of register ENABLE_C3
INIT_ENABLE_C4	int	0xffffffff	initial value of register ENABLE_C4
INIT_ENABLE_C5	int	0xffffffff	initial value of register ENABLE_C5
INIT_ENABLE_C6	int	0xffffffff	initial value of register ENABLE_C6
INIT_ENABLE_C7	int	0xffffffff	initial value of register ENABLE_C7
INIT_ENABLE_D0	int	0xffffffff	initial value of register ENABLE_D0
INIT_ENABLE_D1	int	0xffffffff	initial value of register ENABLE_D1
INIT_ENABLE_D2	int	0xffffffff	initial value of register ENABLE_D2
INIT_ENABLE_D3	int	0xffffffff	initial value of register ENABLE_D3
INIT_ENABLE_D4	int	0xffffffff	initial value of register ENABLE_D4
INIT_ENABLE_D5	int	0xffffffff	initial value of register ENABLE_D5
INIT_ENABLE_D6	int	0xffffffff	initial value of register ENABLE_D6
INIT_ENABLE_D7	int	0xffffffff	initial value of register ENABLE_D7
INIT_MATCH_C0	int	0x0	initial value of register MATCH_C0
INIT_MATCH_C1	int	0x1	initial value of register MATCH_C1
INIT_MATCH_C2	int	0x2	initial value of register MATCH_C2
INIT_MATCH_C3	int	0x3	initial value of register MATCH_C3
INIT_MATCH_C4	int	0x4	initial value of register MATCH_C4
INIT_MATCH_C5	int	0x5	initial value of register MATCH_C5

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-437 Parameters for PL390_GIC (continued)

Name	Type	Default value	Description
INIT_MATCH_C6	int	0x6	initial value of register MATCH_C6
INIT_MATCH_C7	int	0x7	initial value of register MATCH_C7
INIT_MATCH_D0	int	0x0	initial value of register MATCH_D0
INIT_MATCH_D1	int	0x1	initial value of register MATCH_D1
INIT_MATCH_D2	int	0x2	initial value of register MATCH_D2
INIT_MATCH_D3	int	0x3	initial value of register MATCH_D3
INIT_MATCH_D4	int	0x4	initial value of register MATCH_D4
INIT_MATCH_D5	int	0x5	initial value of register MATCH_D5
INIT_MATCH_D6	int	0x6	initial value of register MATCH_D6
INIT_MATCH_D7	int	0x7	initial value of register MATCH_D7
NUM_CPU	int	0x8	number of cpu interfaces
NUM_LSPI	int	0x1f	number of lockable shared peripheral interrupts
NUM_PPI	int	0x10	number of peripheral interrupts
NUM_PRIORITY_LEVELS	int	0x100	number of priority levels
NUM_SGI	int	0x10	number of software generated interrupts
NUM_SPI	int	0x3dc	number of shared peripheral interrupts

3.9.60 PPUMTWakerequest

Power Policy Unit (PPU) v8.2 Multi-threaded Core Wakerequest Logic. This model is written in C++.

Table 3-438 Ports

Name	Protocol	Type	Description
cpu_pchannel_m	*PChannel* on page 2-80	Master	-
ppu_pchannel_s	*PChannel* on page 2-80	Slave	-
thread_wake_request[2]	*Signal* on page 2-85	Slave	-
wakerequest	*Signal* on page 2-85	Master	-

3.9.61 PPUv0

Power Policy Unit (PPU) v0.8 architectural model. This model is written in C++.

PPUv0 contains the following CADI targets:

- PPUv0

PPUv0 contains the following MTI components:

- *4.98 PPUv0 - trace* on page 4-2747
- *4.107 PVBusSlave - trace* on page 4-2760

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-439 Ports

Name	Protocol	Type	Description
irq	*Signal* on page 2-85	Master	-
powerdown	*Signal* on page 2-85	Master	-
ppuhwstat	*Value* on page 2-85	Master	-
pvbus_s	PVBus	Slave	-
reset_in	*Signal* on page 2-85	Slave	-
smpen	*Signal* on page 2-85	Slave	-
standbywfi	*Signal* on page 2-85	Slave	-
wakerequest	*Signal* on page 2-85	Slave	-

3.9.62 PPUv1

Power Policy Unit (PPU) v1.1 architectural model. This model is written in C++.

PPUv1 contains the following CADI targets:

- PPUv1

PPUv1 contains the following MTI components:
- *4.99 PPUv1 - trace* on page 4-2748
- *4.107 PVBusSlave - trace* on page 4-2760

About PPUv1

Software can determine which features the PPU supports by reading the PPU Identification Register 0, PPU_IDR0 and the PPU Identification Register 1, PPU_IDR1.

The following power policies are offered by the PPU model, in order of increasing priority:
- Off.
- Emulated Off.
- Memory Retention.
- Emulated Memory Retention.
- Logic Retention.
- Full Retention.
- Memory Off.
- Functional Retention.
- On.
- Warm Reset.
- Debug Recovery Reset.

For the power mode transition rules, see *Arm Power Policy Unit Architecture Specification*.

There are 16 operating mode values. The meaning of these values is specific to the device that is connected to the PPU. The operating mode can only be configured to change during a power transition of ON to ON.

The PPU model supports static and dynamic transitions on the P-Channel interface. It does not yet support Q-Channel.

DEVPACTIVE and DEVPSTATE have the following bit encodings:

DEVPACTIVE bits [10:0]
Each bit indicates a required power mode.

DEVPSTATE bits [3:0]
The integer formed by this bitfield indicates a power mode.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DEVPACTIVE bits [23:16]
Operating mode. The interpretation of these bits depends on the DEVPACTIVE use model (Ladder or Independent).

DEVPSTATE bits [7:4]
The integer formed by this bitfield indicates an operating mode.

Communication over the Low Power Interface (PREQUEST and PACTIVE) uses blocking calls and does not model any delays. See *2.5.1 PChannel protocol* on page 2-80 for further details.

For static transitions, software sets the policy as the required power mode. The PPU then sends a PREQUEST with the required power state to the attached device. The device can ACCEPT or DENY it. For dynamic transitions, software sets the policy as a minimum power mode. Based on whether the device has sent a signal using DEVPACTIVE, the PPU sends a PREQUEST with the required power state to the attached device. The device can ACCEPT or DENY it.

The PPU model is automatically reset by the simulation engine when the model starts up. Reset can also occur through the reset_in port. The PPU model is reset only when the signal value is Signal::Set. Use Signal::Set instead of zero, its integer value, to prevent unexpected behavior.

The ppuhwstat port notifies the power state change inside the PPU and the definition of each bit is the same as DEVPACTIVE[10:0].

The smpen and standbywfi ports are defined in PPUv0 and are not supported in PPUv1.

Differences between the model and the RTL

- Q-Channel is not supported
- The PPU model has been validated with devices supporting only ON and OFF power modes. Arm has not tested the case where a connected device supports other power modes offered by the PPU.

Table 3-440 Ports

Name	Protocol	Type	Description
devpactive	*PChannel* on page 2-80	Master	P-Channel port
irq	*Signal* on page 2-85	Master	PPU IRQ signal
powerdown	*Signal* on page 2-85	Master	Notify whether or not the PPU is in OFF state.
ppuhwstat	*Value* on page 2-85	Master	Notify the power state change inside the PPU. The definition of each bit is the same as DEVPACTIVE[10:0].
pvbus_s	PVBus	Slave	PPU APB bus slave port
reset_in	*Signal* on page 2-85	Slave	PPU reset signal input
wakerequest	*Signal* on page 2-85	Slave	Input port for the wakerequest signal. It is ORed with PACTIVE[8] (ON) inside the PPU as input to PPU DEVPACTIVE[8] (ON). The "is_core_ppu" parameter controls whether there is additional logic to hold this signal until the PPU is in OFF/OFF_EMU state.

Table 3-441 Parameters for PPUv1

Name	Type	Default value	Description
RevD_support	bool	0x0	Whether to support Rev D locked IRQ
default_op_dyn_en	bool	0x0	Whether to enable operating mode dynamic transition by default
default_op_policy	int	0x0	Default operating policy
default_power_state_on	bool	0x0	Default power state ON
default_pwr_dyn_en	bool	0x0	Whether to enable dynamic power mode transition by default

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-441 Parameters for PPUv1 (continued)

Name	Type	Default value	Description
device_channels	int	0x0	Number of device channels (0: P-Channel, 1-8: Q-Channels)
dynamic_off	bool	0x0	Dynamic Off
dynamic_on	bool	0x0	Dynamic On
dynamic_warm_reset	bool	0x0	Dynamic Warm Reset
full_ret	int	0x0	Full Retention (0: not supported, 1: static, 2: dynamic)
func_ret	int	0x0	Functional Retention (0: not supported, 1: static, 2: dynamic)
is_core_ppu	bool	0x0	PPU is core_ppu type which means wake_request would wait till PPU is OFF/OFF_EMU
lock_support	bool	0x1	Whether to support OFF lock feature
logic_ret	int	0x0	Logic Retention (0: not supported, 1: static, 2: dynamic)
mem_off	int	0x0	Memory Off (0: not supported, 1: static, 2: dynamic)
mem_ret	int	0x0	Memory Retention (0: not supported, 1: static, 2: dynamic)
num_opmode_cfg	int	0x0	Number of operating modes
op_active_cfg	int	0x0	Operating mode active configuration (0: Ladder use model, 1: Independent user model)
revision	string	"r1p1"	Revision
use_active_signal	bool	0x0	Use device-active signal

Related information

AMBA Low Power Interface Specification ARM Q-Channel and P-Channel Interfaces (IHI 0068C ID091216)

3.9.63 PPUv1_Cluster_Wakerequest_Logic

PPUv1 wake request stall logic. This model is written in C++.

Table 3-442 Ports

Name	Protocol	Type	Description
cluster_wake_request	*Signal* on page 2-85	Master	-
core_wake_request_in[8]	*Signal* on page 2-85	Slave	-
core_wake_request_out[8]	*Signal* on page 2-85	Master	-
ppuhwstat	*Value* on page 2-85	Slave	-
reset_in	*Signal* on page 2-85	Slave	-

3.9.64 SMMUv3AEM

This model is written in C++.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SMMUv3AEM contains the following CADI targets:

- SMMUv3AEM

SMMUv3AEM contains the following MTI components:

- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760
- *4.111 SMMUv3AEM - trace* on page 4-2770

About SMMUv3AEM

The SMMUv3 Architecture Envelope Model component is an architectural model that implements the SMMUv3.0, SMMUv3.1, and SMMUv3.2 architectures for I/O virtualization of devices, except for the limitations listed below.

The SMMUv3 specifies that input addresses are conceptually 64 bits. The SMMUv3AEM model assumes that the input address is 64 bits. If the SoC has less than 64 bits as an input address bus then if the SoC wants to use the high address space (and use TT1) then it must sign extend the address from the upstream peripherals to get to 64 bits.

The SMMUv3AEM model deals with groups of transactions with the same attributes and a similar range of addresses. The mapping used is remembered by the bus infrastructure and is used for subsequent sufficiently similar transactions without requiring intervention from the SMMU model, and so will not be traced, for instance. The range of addresses that the mapping is valid for is determined by the SMMU model, depending on architectural and model-implementation details. However, as it is unaware of any sign-extension then the mapping that the SoC does, the SoC is responsible for subsequently narrowing the range of addresses for which this mapping is valid. Typically, this is done automatically when using `PVBusMapper`.

The model has the following limitations:

- It does not support:
  - RAS.
  - Power control.
  - AMBA stash operations and destructive read operations are not supported on `PVBus` and also are not supported by the device.
  - PCIe-NoSnoop transactions.
- The PMU has limited functionality. Only a subset of the architecturally mandatory events are supported, as indicated by the `SMMU_PMCG_CEID0` fields. The PMU is intended for demonstration purposes only and for driver development.

Notes

- A bad configuration renders the model inactive.
- Some configurations can be adjusted by configuration pins. These are only sampled at the negative edge of the reset pin. If you want to use these pins then you must drive them before sending a negative edge on the reset pin. During simulation_reset the component driving them must also drive this transition again.
- Debug reads to the registers do not disturb the state.
- Writes to registers with Update flags, including debug writes, are ignored if the Update flag is already set to one.
- Debug and real accesses to the registers must be 32 bits or 64 bits.
- MSIs are issued using attributes determined by the parameter `msi_attribute_transform`, whilst Event queue writes are always issued with `ExtendedID=0`, `UserFlags=0`, `MasterID=0xFFFFffff`. Thus, if your system needs to distinguish MSI writes from Event queue writes, it can do so using this mechanism.
- If your system does table walks and queue accesses through TBU0 (`separate_tw_msi_qs_port == false`), then care must be taken to distinguish table walk and queue traffic (with `MasterID=0xFFFFffff`) from normal translated traffic.
- If `SMMU_IDR1.TABLES_PRESET` or `SMMU_IDR1.QUEUES_PRESET` is set then see parameter `PRESET_REL_base_address` and the parameters it mentions. Embedded implementations of the

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SMMU are allowed to have the queues/stream table in a 'close' RAM, either on-chip or in the SMMU itself. For the model, it is up to the integrator to supply this memory and for the SMMU model to be able to access. Thus if the actual hardware has the memory built into the SMMU then it will be necessary for the integrator to wrap this model with a bus decoder and a memory model to more closely model the embedded implementation.

Table 3-443 Ports

Name	Protocol	Type	Description
clk_in	ClockSignal	Slave	Clock signal
conf_reset_of_SMMU_GBPA_ABORT	*Signal* on page 2-85	Slave	System reset value of SMMU_GBPA.ABORT.
conf_reset_of_SMMU_S_GBPA_ABORT	*Signal* on page 2-85	Slave	System reset value of SMMU_S_GBPA.ABORT.
conf_system_supports_btm	*Signal* on page 2-85	Slave	System supports BTM and will be reflected in the IDR registers. This signal can override the value set by the parameters configuring the IDR registers. If BTM (Broadcast Table Maintenance) is not supported then DVM messages will be ignored.
conf_system_supports_cohacc	*Signal* on page 2-85	Slave	System supports COHACC and will be reflected in the IDR registers. This signal can override the value set by the parameters configuring the IDR registers. If COHACC is set then page walks and SMMU-generated accesses will have the required shareability set, otherwise they will be marked as non-shareable.
conf_system_supports_httu	*Signal* on page 2-85	Slave	System supports HTTU and will be reflected in the IDR registers. See parameter support_for_httu_when_starts_disallowed for the use of this signal.
conf_system_supports_sev	*Signal* on page 2-85	Slave	System supports SEV and will be reflected in the IDR registers. This signal can override the value set by the parameters configuring the IDR registers.
identify	*SMMUv3AEMIdentifyProtocol* on page 2-78	Master	Map the transaction to the tuple (StreamID, SubstreamID, SubstreamIDValid, SSD)
irq_out_command_queue_sync_ns	*Signal* on page 2-85	Master	Pulsed interrupt output signal for non-secure CMD_SYNC having a completion signal of SIG_IRQ.
irq_out_command_queue_sync_s	*Signal* on page 2-85	Master	Pulsed interrupt output signal for secure CMD_SYNC having a completion signal of SIG_IRQ.
irq_out_event_queue_ns	*Signal* on page 2-85	Master	Pulsed interrupt output signal for the non-secure event queue becoming non-empty.
irq_out_event_queue_s	*Signal* on page 2-85	Master	Pulsed interrupt output signal for the secure event queue becoming non-empty.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-443 Ports (continued)

Name	Protocol	Type	Description
irq_out_gerror_ns	*Signal* on page 2-85	Master	Pulsed interrupt output signal for non-secure SMMU_GERROR(N) signalling an error.
irq_out_gerror_s	*Signal* on page 2-85	Master	Pulsed interrupt output signal for secure SMMU_GERROR(N) signalling an error.
irq_out_ns	*Signal* on page 2-85	Master	Pulsed interrupt output signal combined from all non-secure (non-RAS) interrupts.
irq_out_pmcg_ns_as_value	*Value* on page 2-85	Master	Non-secure PMCG interrupt value port. Value port representing a set of Performance Monitor Counter Group interrupts. There is an unknown number of PMCGs and so an unknown number of PMCG interrupts. There may not necessarily even be an interrupt per group. We export the interrupt to be generated as a unsigned. The 'pmcg_index' is exported on the top 16 bits and the 'pmcg_counter' (that overflowed) on the bottom 16 bits. Any configured MSI for this group will be the next transaction out of the pvbus_m_tw_msi_qs port. The architecture supports an MSI from a PMCG could come out of a different port (say the TBU master port that a PMCG might be associated with). However the AEM only supports it coming out of the TCU port (pvbus_m_tw_msi_qs).
irq_out_pmcg_s_as_value	*Value* on page 2-85	Master	Secure PMCG interrupt value port. Value port representing a set of Performance Monitor Counter Group interrupts. There is an unknown number of PMCGs and so an unknown number of PMCG interrupts. There may not necessarily even be an interrupt per group. We export the interrupt to be generated as a unsigned. The 'pmcg_index' is exported on the top 16 bits and the 'pmcg_counter' (that overflowed) on the bottom 16 bits. Any configured MSI for this group will be the next transaction out of the pvbus_m_tw_msi_qs port. The architecture supports an MSI from a PMCG could come out of a different port (say the TBU master port that a PMCG might be associated with). However the AEM only supports it coming out of the TCU port (pvbus_m_tw_msi_qs).
irq_out_pri_queue	*Signal* on page 2-85	Master	Pulsed interrupt output signal for the PRI queue.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-443 Ports (continued)

Name	Protocol	Type	Description
irq_out_ras_cri_as_value	*Value* on page 2-85	Master	RAS Critical error interrupt value port. The number of RAS error record groups is dependent on the configuration. As each error record group could have its own interrupts then we write to this value port the error record group number (which is dependent on the configuration) to simulate an interrupt pulse. We only support modelling edge-triggered interrupts. The value passed is formatted as follows: bits[23:16] the ras error group index (if multiple error groups) (0 atm) bits[15:8] node id bits[7:0] the ras record index in the node.
irq_out_ras_eri_as_value	*Value* on page 2-85	Master	RAS Error Handling Interrupt value port. The number of RAS error record groups is dependent on the configuration. As each error record group could have its own interrupts then we write to this value port the error record group number (which is dependent on the configuration) to simulate an interrupt pulse. We only support modelling edge-triggered interrupts. The value passed is formatted as follows: bits[23:16] the ras error group index (if multiple error groups) (0 atm) bits[15:8] node id bits[7:0] the ras record index in the node.
irq_out_ras_fhi_as_value	*Value* on page 2-85	Master	RAS Fault Handling Interrupt value port. The number of RAS error record groups is dependent on the configuration. As each error record group could have its own interrupts then we write to this value port the error record group number (which is dependent on the configuration) to simulate an interrupt pulse. We only support modelling edge-triggered interrupts. The value passed is formatted as follows: bits[23:16] the ras error group index (if multiple error groups) (0 atm) bits[15:8] node id bits[7:0] the ras record index in the node.
irq_out_s	*Signal* on page 2-85	Master	Pulsed interrupt output signal combined from all secure (non-RAS) interrupts.
pvbus_control_s	PVBus	Slave	Register slave port

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-443 Ports (continued)

Name	Protocol	Type	Description
pvbus_id_routed_m	PVBus	Master	This is a special "id-routed" port for transmitting ATC invalidates and PRI Responses upstream into the PCIe EndPoints, it is not a normal bus. The FastSim ATC invalidate protocol specifies how to route and deal with this this port. It is assumed that the StreamID can uniquely route the transaction if there are multiple PCIe Root Complexes.
pvbus_m[64]	PVBus	Master	The TBU master ports that carry transactions that have been translated from the correspondingly numbered pvbus_s[] port.
pvbus_m_tw_msi_qs	PVBus	Master	Master port used for Table Walks, MSIs and Queue access when separate_tw_msi_qs_port==true
pvbus_s[64]	PVBus	Slave	The TBU slave ports that receive transactions to be translated. They will exit the SMMU through the same numbered pvbus_m[] port.
reset_in	*Signal* on page 2-85	Slave	Reset signal
sev_out	*Signal* on page 2-85	Master	Event signal

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-444 Parameters for SMMUv3AEM

Name	Type	Default value	Description
PRESET_REL_base_address	int	0x0	If using preset addresses (SMMU_IDR1.QUEUES_PRESET/ TABLES_PRESET) then the queue and table base registers become fixed. If SMMU_IDR1.REL then the addresses are relative to the base of the register file and this parameter tells the model what address to add to the queue/table addresses to calculate the actual address. This is for 'embedded implementations' where the memory for these structures is held within the SMMU itself or in a 'close' RAM. The model does not contain any RAM and the integrator must supply a RAM at the appropriate address. If the preset tables/queues overlap, the RAM has to implement separate secure and non-secure address spaces. See also: * TABLES_PRESET_smmu_strtab_base * TABLES_PRESET_smmu_strtab_base_cfg * TABLES_PRESET_smmu_s_strtab_base * TABLES_PRESET_smmu_s_strtab_base_cfg * QUEUES_PRESET_smmu_cmdq_base * QUEUES_PRESET_smmu_s_cmdq_base * QUEUES_PRESET_smmu_eventq_base * QUEUES_PRESET_smmu_s_eventq_base * QUEUES_PRESET_smmu_priq_base
QUEUES_PRESET_smmu_cmdq_base	int	0x0	If SMMU_IDR1.QUEUES_PRESET == 1 then this is the value that appears in SMMU_CMDQ_BASE and SMMU_CMDQ_BASE becomes read-only. See also parameter PRESET_REL_base_address.
QUEUES_PRESET_smmu_eventq_base	int	0x0	If SMMU_IDR1.QUEUES_PRESET == 1 then this is the value that appears in SMMU_EVENTQ_BASE and SMMU_EVENTQ_BASE becomes read-only. See also parameter PRESET_REL_base_address.
QUEUES_PRESET_smmu_priq_base	int	0x0	If SMMU_IDR1.QUEUES_PRESET == 1 and SMMU_IDR0.PRI == 1 then this is the value that appears in SMMU_PRIQ_BASE and SMMU_PRIQ_BASE becomes read-only. See also parameter PRESET_REL_base_address.
QUEUES_PRESET_smmu_s_cmdq_base	int	0x0	If SMMU_IDR1.QUEUES_PRESET == 1 and SMMU_S_IDR1.SECURE_IMPL == 1 then this is the value that appears in SMMU_S_CMDQ_BASE and SMMU_S_CMDQ_BASE becomes read-only. See also parameter PRESET_REL_base_address.
QUEUES_PRESET_smmu_s_eventq_base	int	0x0	If SMMU_IDR1.QUEUES_PRESET == 1 and SMMU_S_IDR1.SECURE_IMPL == 1 then this is the value that appears in SMMU_S_EVENTQ_BASE and SMMU_S_EVENTQ_BASE becomes read-only. See also parameter PRESET_REL_base_address.
SMMU_AIDR	int	0x0	SMMU_AIDR contains the Major and Minor architectural revisions numbers
SMMU_IDR0	int	0x0	SMMU_IDR0. The following fields are further combined with the port conf_system_supports_{sev,httu,btm,cohacc}:- sev httu btm cohacc

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-444 Parameters for SMMUv3AEM (continued)

Name	Type	Default value	Description
SMMU_IDR1	int	0x0	SMMU_IDR1.
SMMU_IDR2	int	0x0	SMMU_IDR2 holds the BA_VATOS field.
SMMU_IDR3	int	0x0	SMMU_IDR3 is reserved.
SMMU_IDR4	int	0x0	SMMU_IDR4 is Imp def.
SMMU_IDR5	int	0x0	SMMU_IDR5 contains, amongst others the output address encoded size (OAS).
SMMU_IDR6	int	0x0	SMMU_IDR6 is RES0 if Enhanced Command Queues do not exist (SMMU_IDR1.ECMDQ == 0). Otherwise, SMMU_IDR6 contains information about the configuration of the ECMDQs.
SMMU_IIDR	int	0x0	SMMU_IIDR contains fields for the implementer, product revision, etc.
SMMU_MPAMIDR	int	0x0	SMMUv3.2: If SMMU_IDR3.MPAM == 1 then SMMU_MPAMIDR holds further ID information for Memory Partitioning And Monitoring (MPAM) extension. This is optional in SMMUv3.2 and is backported to SMMUv3.1.
SMMU_S_IDR0	int	0x0	Secure IDR0 register.
SMMU_S_IDR1	int	0x0	SMMU_S_IDR1 Indicates if there is a secure side by bit 31.
SMMU_S_IDR2	int	0x0	SMMU_S_IDR2 Reserved
SMMU_S_IDR3	int	0x0	SMMU_S_IDR3 Reserved
SMMU_S_IDR4	int	0x0	SMMU_S_IDR4 IMP DEF
SMMU_S_IDR6	int	0x0	SMMU_S_IDR6 is RES0 if Secure Enhanced Command Queues do not exist (SMMU_S_IDR0.ECMDQ == 0). Otherwise, SMMU_S_IDR6 contains information about the configuration of the ECMDQs.
SMMU_S_MPAMIDR	int	0x0	SMMUv3.2: If SMMU_IDR3.MPAM == 1 then SMMU_S_MPAMIDR holds further ID information for Memory Partitioning And Monitoring (MPAM) extension. This is optional in SMMUv3.2 and is backported to SMMUv3.1.
TABLES_PRESET_smmu_s_strtab_base	int	0x0	If SMMU_IDR1.TABLES_PRESET == 1 and SMMU_S_IDR1.SECURE_IMPL == 1 then this is the value that appears in SMMU_S_STRTAB_BASE and SMMU_S_STRTAB_BASE becomes read-only. See also parameter PRESET_REL_base_address.
TABLES_PRESET_smmu_s_strtab_base_cfg	int	0x0	If SMMU_IDR1.TABLES_PRESET == 1 and SMMU_S_IDR1.SECURE_IMPL == 1 then this is the value that appears in SMMU_S_STRTAB_BASE_CFG and SMMU_S_STRTAB_BASE_CFG becomes read-only. See also parameter PRESET_REL_base_address.
TABLES_PRESET_smmu_strtab_base	int	0x0	If SMMU_IDR1.TABLES_PRESET == 1 then this is the value that appears in SMMU_STRTAB_BASE and SMMU_STRTAB_BASE becomes read-only. See also parameter PRESET_REL_base_address.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-444 Parameters for SMMUv3AEM (continued)

Name	Type	Default value	Description
TABLES_PRESET_smmu_strtab_base_cfg	int	0x0	If SMMU_IDR1.TABLES_PRESET == 1 then this is the value that appears in SMMU_STRTAB_BASE_CFG and SMMU_STRTAB_BASE_CFG becomes read-only. See also parameter PRESET_REL_base_address.
all_error_messages_through_trace	bool	0x0	Some conditions in the SMMU are so strange that the software programming the SMMU has done something wrong. At this point messages are output to either ArchMsg.Error.* or ArchMsg.Warning.* or to the error stream of the simulator. Outputting to the error stream of the simulator may cause it to return with a non-zero exit status. If you set this option to true then instead of using the error stream of the simulator it will always use a trace stream allowing the simulation to exit with a zero exit status.
allow_non_secure_access_to_SMMU_S_INIT	bool	0x0	If the system has no software operating as a secure agent then set this parameter. This allows non-secure accesses to the SMMU_S_INIT register and allows the non-secure software to reset the TLB, clearing out any 'secure' TLB entries. If the SMMU does not implement the security extensions (SMMU_S_IDR1.SECURE_IMPL == 0) then this parameter is ignored.
ats_split_stage_dbm_update_do_with_ATSRequest	int	0x0	When doing split-stage ATS, then the DBM update for the final stage 2 descriptor can be done either whilst processing the ATS request or delayed until it actually sees the PCIe Translated Transaction using the stage 2 descriptor. 0 -- do when see actual transaction 1 -- do when processing the ATS request 2 -- do it randomly with 50% chance.
behaviour_of_sampled_at_reset_signals	int	0x0	Some configuration signals into the SMMU are sampled on negedge of reset. However, it can sometimes be hard to arrange to drive a configuration pin before the negedge of reset. The configuration pins are sampled: 0 -- at negedge reset. 1 -- at negedge reset, but if a later change occurs at the same simulated time, and no transactions have occurred, then they will be resampled and the SMMU reset again.
cmdq_max_number_of_commands_to_buffer	int	0xa	The command queues can buffer fetched commands and commands before issuing them. This parameter is roughly the maximum number of commands to do this for. The programmer visible effects are that just because the CONS pointer shows a command has been _consumed_ does not necessarily mean that it has been issued (and completed). Higher values will accentuate this effect.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-444 Parameters for SMMUv3AEM (continued)

Name	Type	Default value	Description
enable_device_id_checks	bool	0x1	If this parameter is true then the DeviceIDs seen by the GIC are: * for client devices: DeviceID = StreamID + translated_device_id_base * for SMMU-generated MSIs: smmu_msi_device_id This parameter enables two checks: * If the DeviceID is used in the output_attribute_transform then if it overflows 32 bits then the model will warn. If the DeviceID is not used then it is assumed that the external agent that forms the DeviceID will warn if it overflows. * If the SMMU supports MSIs, then the model will check that the GIC will be able to distinguish an MSI generated by the SMMU from one generated by a client device. As the exact mechanism to determine the DeviceID is in the system and not necessarily under control of the SMMU then you can disable these warnings using this parameter. See also the parameters: output_attribute_transform and msi_attribute_transform.
howto_identify	string	"use-identify"	If 'use-identify' then will use the 'identify' port to determine the SSD, StreamID, SubStreamID of a transaction. Otherwise, this string extracts them from the transaction's attributes. Examples:- SEC_SID=ExtendedID[63], SSV=ExtendedID[62], SubstreamID=ExtendedID[51:32], StreamID=ExtendedID[31:0] nSEC_SID=ExtendedID[63], StreamID=ExtendedID[55:24], nSSV=ExtendedID[20], SubstreamID=ExtendedID[19:0] SEC_SID is one bit wide, true if StreamID is secure. SSV is one bit wide, true if SubstreamID is valid. The alternative symbols nSEC_SID and nSSV have the negative logic to SEC_SID and SSV. You must not use the negative and positive logic for the same symbol at the same time. However, just because you use negative logic for one symbol does not force you to use negative logic for the other. SubstreamID is 20 bits wide and StreamID is 32 bits. More complex examples: StreamID[31:24]=0, StreamID[23:0]=ExtendedID[23:0], SSV=1[0], ... Available attributes: ExtendedID, MasterID, UserFlags
httu_early_st2_permission_fault_if_af_update_at_stage1	int	0x0	If a stage 1 descriptor needs an HTTU update, but the descriptor is unwriteable at stage 2 and also a stage 1 permission fault occurs, then the architecture permits either the stage 1 or stage 2 permission fault to be recorded. 0 -- stage 1 permission check check stage 1 descriptor writeable at stage 2 if AF- or DBM-update required 1 -- check stage 1 descriptor writeable at stage 2 if AF-update required stage 1 permission check check stage 1 descriptor writeable at stage 2 if DBM-update required 2 -- do behaviour 1 or 2 randomly with a 50% chance.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-444 Parameters for SMMUv3AEM (continued)

Name	Type	Default value	Description
`httu_memory_types_supported`	`string`	"rawaWB, raWB, waWB, naWB"	This is a comma-separated list of memory types that are IMPLEMENTATION DEFINED as supporting HTTU. However, the system must have Far Atomic support for the specified memory address and memory type. Device types: * nGnRnE, nGnRE, nGRE, GRE Normal memory types are composed of an 'inner' and an 'outer' cacheability. The model only supports types where the inner and outer are identical. * Normal non-cacheable types * nc_nb, nc * Cacheable types are of the form (na?|(ra)?(wa)?)(WT|WB)(tr)? * na/ra/wa -- no/read/write allocate * WT/WB -- write through/write back * tr -- transient * exceptions: * 'na' and 'tr' are incompatible * without 'na' then you must specify at least one of ra/wa. Example: "WT" is illegal, "raWT" is legal. rawaWB is always supported and it is optional. Examples: * "rawaWB, raWB, waWB, naWB" -- only the WB type is supported * "nc" -- rawaWB and the normal non-cacheable type are supported
`imp_def_L1CD_L2Ptr_out_of_range`	`int`	`0x0`	If an L1CD.L2Ptr is out of range of IAS/OAS as appropriate then what happens is controlled by this parameter: 0 -- if is an IPA, then Stage 2 Translation Fault, if is a PA then truncate to OAS 1 -- generate C_BAD_SUBSTREAMID if an IPA and > IAS, or if a PA and > OAS. 2 -- generate C_BAD_SUBSTREAMID if an IPA and > IAS, or F_CD_FETCH if a PA and > OAS. 3 -- truncate the IPA or PA to IAS/OAS as appropriate NOTE that if the model is configured as SMMUv3.1 then this parameter is IGNORED, and behaves as though this parameter was set to 1. The SMMUv3.1 architecture actually allows more behaviours but the model will only implement this one. NOTE that the SMMUv3.0 allows more behaviours than can be expressed by this parameter.
`imp_def_PID0`	`int`	`0x83`	If imp_def_has_PID_CID is true then this is the PID0 value.
`imp_def_PID1`	`int`	`0xb4`	If imp_def_has_PID_CID is true then this is the PID1 value.
`imp_def_PID2`	`int`	`0xb`	If imp_def_has_PID_CID is true then this is the PID2 value.
`imp_def_PID3`	`int`	`0x0`	If imp_def_has_PID_CID is true then this is the PID3 value.
`imp_def_PID4`	`int`	`0x4`	If imp_def_has_PID_CID is true then this is the PID4 value.
`imp_def_S1ContextPtr_out_of_range`	`int`	`0x0`	If an STE is fetched that uses a stage 1 then if: - Stage 1 only and S1ContextPtr > OAS, or - Stage 1+2 and S1ContextPtr > IAS then what happens is IMP DEF and this parameter controls the behaviour:- 0 -- stage 1 only -- C_BAD_STE -- stage 1+2 -- C_BAD_STE 1 -- stage 1 only -- C_BAD_STE -- stage 1+2 -- truncate to IAS 2 -- stage 1 only -- truncate to OAS -- stage 1+2 -- C_BAD_STE 3 -- stage 1 only -- truncate to OAS -- stage 1+2 -- truncate to IAS 4 -- stage 1 only -- truncate to OAS -- stage 1+2 -- Stage 2 translation fault 5 -- stage 1 only -- C_BAD_STE -- stage 1+2 -- Stage 2 translation fault The architecture also allows for F_CD_FETCH, but the model does not support this. NOTE that in SMMUv3.1 then the only allowed values of this parameter are 0 or 5.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-444 Parameters for SMMUv3AEM (continued)

Name	Type	Default value	Description
imp_def_alloccfg	int	0x0	ALLOCCFG overrides the read/write/transient hints on cacheable types. However these are hints and an implementation may choose to treat them differently. 0 -- apply the alloc hints as architecturally specified 1 -- ignore all ALLOCCFG fields (treated as zero)
imp_def_ats_attribute_stashing	int	0x0	The SMMU architecture allows an ATS request to return the attributes with which to make the Translated Access. PCIe does not define any transaction attributes in the ARM sense and so the mechanism for doing this is IMP DEF. Usually this would be done by packing them into the high order address bits of the return response. In the model, then the representation of the ATS reply returns the attributes directly and it is up to the ATC whether it wants to use them or not. The parameter configures what to place in those architectural attributes in the ATS Reply. 0 -- the architectural attributes 1 -- Inner Write Back, Outer Write Back, Inner Shared, read and write allocate, User-Data 2 -- Inner Write Back, Outer Write Back, Outer Shared, read and write allocate, User-Data The SMMU cannot force an ATC to use these attributes, it is simply the attributes that are returned in the non-PCIe part of the ATS reply.
imp_def_cohacc_effect	int	0x0	SMMU_IDR0.COHACC is a system property. However, the exact nature of the transactions that the SMMU emits is an IMP DEF property when COHACC == 0: 0 -- COHACC == 0 forces the output attributes of SMMU-generated accesses to non-shared. 1 -- The only effect of COHACC is what is reported in SMMU_IDR0.COHACC and has no effect on the output attributes of SMMU-generated accesses.
imp_def_effective_ATTR_TYPES_OVR_is_false_per_port	string	""	SMMU_IDR1.ATTR_TYPES_OVR == 1 means that the STE and SMMU_(S_)GBPA MTCFG/SHCFG/ALLOCCFG have an effect. However, an implementation is allowed to ignore this being one for specific ports and _not_ apply the overrides MTCFG/SHCFG/ALLOCCFG despite SMMU_IDR1.ATTR_TYPES_OVR == 1. This parameter is a comma-separated lists of port ranges (indexed from 0) for those ports where SMMU_IDR1.ATTR_TYPES_OVR behaves as 0. For example: 0, 10-20, 40
imp_def_has_PID_CID	bool	0x1	If this is true then the SMMU model will have the standard PID/CID ID registers. Only the PID0..PID4 registers can be customized and the parameters imp_def_PID0..imp_def_PID4 are used.
imp_def_ns_bit_for_s_vatos_on_s1_bypass	int	0x0	When a Secure VATOS operations for a translation that bypasses stage 1 by S1DSS then the output NS bit is the same as the input NS bit of the translation. The architecture does not provide an input NS bit in the SMMU_S_VATOS_ADDR register and it is treated as an IMP DEF value. This parameter specifies that value: 0 -- secure 1 -- non-secure 2 -- random

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-444 Parameters for SMMUv3AEM (continued)

Name	Type	Default value	Description
imp_def_ras_allow_non_secure_accesses_if_supports_secure	bool	0x0	If two security worlds are supported, i.e.: SMMU_S_IDR1.SECURE_IMPL == 1 then if this parameter is true, then non-secure accesses are allowed to access any RAS registers (see parameter 'ras'). Otherwise, non-secure accesses are RAZ/WI. If only a single security state (non-secure) is supported, then this parameter is ignored and non-secure accesses are always allowed.
imp_def_reset_unknown_fields_to_zero	bool	0x0	Many fields and registers in the SMMUv3 architecture reset to an UNKNOWN value. However, many implementations will choose to reset to 0. By setting this parameter to true then those fields will be initialised to zero.
imp_def_split_ATS_attributes_is_stage1	bool	0x0	If using split stage ATS, then it is IMP DEF whether the stage 1 attributes are returned to the ATS request or stage 2. This only has a meaning if the SMMU can stash attributes in the ATS response.
imp_def_truncate_out_of_range_streamids_on_invalidate_commands	bool	0x0	If this parameter is true then the StreamID fields of the following commands will be truncated to (S_)SIDSIZE: * CMD_ATC_INV * CMD_CFGI_STE * CMD_CFGI_STE_RANGE * CMD_CFGI_CD * CMD_CFGI_CD_ALL Otherwise, these commands will NOP.
imp_def_v3_atos_fault	int	0x0	For an IPA to PA ATOS translation that encounters a Stage 1 Address Size Fault then the PAR.REASON field reports: * in SMMUv3.1, 'Stage 1' (0) * in SMMUv3.0, 'Stage 1' (0) or 'Input' (3) depending on the implementation. This parameter is ignored for SMMUv3.1. For SMMUv3.0 then the values are: 0 -- report as 'Input' (3) 1 -- report as 'Stage 1' (0)
mpam_attribute_transform	string	"ExtendedID[62:55]=MPAM_PMG, ExtendedID[54:39]=MPAM_PARTID, ExtendedID[38]=MPAM_NS"	If MPAM is supported, then this is the transform applied to _all_ downstream transactions to transport the MPAM information. If MPAM is not supported then the transform is not applied, however, it is still checked that it is syntactically checked. If MPAM is supported in one security state, it is considered to be supported in all security states, even if that state has PARTID_MAX == 0 or PMG_MAX == 0 and the transform will be applied. NOTE that for translated transactions from client devices then: MPAM_NS = ! SEC_SID

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-444 Parameters for SMMUv3AEM (continued)

Name	Type	Default value	Description
msi_attribute_transform	string	"ExtendedID[31:0]=smmu_msi_device_id, MasterID=0xFFFFffff"	This parameter determines the MSI output attributes of SMMU-generated MSIs: * MasterID * UserFlags * ExtendedID of the transaction to use. This is a set of comma separated transforms on the output attributes. The right hand side is: * a numeric literal (or a slice of a literal) * the parameter smmu_msi_device_id * the symbol 'interrupt_kind' * 0/1 -- EVENTQ Secure/Non-secure * 2 -- PRIQ * 3/4 -- CMD_SYNC Secure/Non-secure * 5/6 -- GERROR Secure/Non-secure * 7/8 -- PMCG Secure/Non-Secure * 9/10/11 -- RAS FHI/ERI/CRI * HWATTR_KIND_0 Example: UserFlags[15:0]=smmu_msi_device_id[31:16], MasterID[15:0]=smmu_msi_device_id[15:0], ExtendedID=0 This transform might be used as part of a system-specific way of determining the DeviceID that is passed to the GIC to distinguish MSIs generated by the SMMU and those generated by client devices of the SMMU. See also: output_attribute_transform and enable_device_id_checks.
msi_ra_wa_tr	int	0x7	A bitmap of the Read Allocation, Write Allocate and Transient hints for MSIs to cacheable memory: bit[0] Transient bit[1] Write Allocate bit[2] Read Allocate If not Write Allocate then it will be forced to Read Allocate as a limitation of AMBA.
non_arch_incoming_stronger_than_iWB_oWB_forces_output_iNC_oNC_or_stronger	string	""	If not empty, then this enables a specific non-architectural behaviour on the comma-separated list of port indexes, or ranges. For example: 0, 10-20, 40 In the normal translation process, then the input attributes are usually replaced by the attributes from the page tables or SMMU_(S_)GBPA. The behaviour is: if incoming attributes are iWB-oWB use the architectural attributes else use the stronger of iNC-oNC-osh and the architectural attributes. This is useful if the ports represent transactions from the PCIe subsystem and the PCIe devices output: * iWB-oWB if not No_Snoop -> output is architectural attributes * iNC-oNC-osh if No_Snoop -> output is iNC-oNC-osh or stronger
number_of_ports	int	0x1	The number of port pairs that the SMMU has.
out_of_range_CMD_ATC_INV_Size	int	0x0	If CMD_ATC_INV.Size > 52 then the model is allowed to:- 0 -- raise CERROR_ILL 1 -- treat as NOP The architecture also allows for an UNKNOWN invalidate size to be used as well but the model does not support this.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-444 Parameters for SMMUv3AEM (continued)

Name	Type	Default value	Description
output_attribute_transform	string	"ExtendedID[31:0]=DeviceID"	Transform the downstream attributes of a transaction. * ""/"none" -- no transform * How to alter the output attributes. An example is: "ExtendedID[15:0]=DeviceID[15:0], UserFlags[31]=nSSV, UserFlags[19:0]=SubstreamID" The attributes that can appear on the left hand side of the transform are ExtendedID, MasterID and UserFlags. The source attributes that can be used are: * ExtendedID/MasterID/UserFlags -- the incoming attributes. * DeviceID -- StreamID + translated_device_id_base * StreamID/SubstreamID/SSV/SEC_SID * nSEC_SID/nSSV -- the negative logic versions. * St1PBHA/St2PBHA -- the Page Based Hardware Attributes from any used leaf descriptors (or zero if not used). * STE_IMPDEF1 -- STE[127:116] * HWATTR_KIND_0 -- PBHA information The right hand side may also contain numeric literals. Any bits of the attributes with no transform are unchanged. NOTE that MPAM information comes from the parameter mpam_attribute_transform.
percent_commit	int	0x14	Percentage of times that a read of a register with Update will commit the update. 0 means commit immediately.
percent_commit_Update_clear	int	0x14	Percentage of times that a read of a register with a pending Update clear will lower the Update flag.
pmu	string	""	What to instantiate as a PMU. NOTE that all events and counters are intended for demonstration purposes only and should not be treated as in any way reflecting accurate values for a real implementation. The model's internal representation of actions differ significantly from real hardware and the particular value obtained from the counters should not be used for benchmarking. Values of this parameter are: * "" -- no PMU * "distributed-0" * "distributed-1" distributed-0: * a PMCG per TBU (number_of_ports, up to 63 ports) * a single PMCG for a TCU * Connect a debugger to see the configuration. distributed-1: * same as distributed-0, except for supporting MSIs and MPAM on the MSIs if MPAM is supported by rest of the SMMU.
ports_that_ignore_PnU_InD_on_transactions_with_no_SubstreamID	string	""	Some bus systems (notably PCIe) do not support marking a transaction as Privileged/User or Instruction/Data unless the transaction has a SubstreamID. This accepts a comma separated list of numbers and ranges, for example: 0, 10-12, 15 If the number P is named in this list then the upstream pvbus_s[P] will have all transactions with no Substream considered to be User and Data.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-444 Parameters for SMMUv3AEM (continued)

Name	Type	Default value	Description
prefetch_only_requests	int	0x0	The simulator supports 'prefetch-only' DMI requests, which can occur for anytime for any reason and are intended to be invisible to the end execution of the model and to the user. 0 -- deny all prefetch-only requests 1 -- use debug requests for any page table walks -- form and use debug TLB/cache entries -- any faults will not record, but deny the prefetch request 2 -- treat prefetch-only requests like normal transactions -- use normal page table walk transactions -- use and form normal TLB/cache entries -- faults will alter the programmer visible state of the SMMU 0 is the safest. 1 treats the access like a debug request and requires that debug page table walks are treated correctly downstream. Any descriptors that need HTTU to allow the transaction to proceed will fail the request. 2 is dangerous, it use real transactions and reports faults that are unphysical. Real transactions can be wait()ed and this disobeys the SystemC spec for get_direct_mem_ptr().
ras	string	""	What to instantiate for RAS handling. Values of this parameter are: * "" -- no RAS records * "mmu600" * "mmu700" mmu600: * only a corrected errors reported. See also imp_def_ras_allow_non_secure_accesses_if_supports_secure.
reset_value_of_SMMU_GBPA	int	0x0	Reset value of SMMU_GBPA
reset_value_of_SMMU_S_GBPA	int	0x0	Reset value of SMMU_S_GBPA
seed	int	0x12345678	Used to seed the pseudo-random number generator that the SMMU model uses.
separate_tw_msi_qs_port	bool	0x1	True if there is a separate port which is used to walk configuration tables, translation tables, issue MSIs and access the queues. If this is false then pvbus_m[0] will be used.
size_of_cd_cache	int	0x0	The number of entries in the cache holding CD structures. If this is zero then it is treated as a large number ('infinite') but it is bounded so that the host memory usage of the cache is also bounded.
size_of_l1cd_cache	int	0x0	The number of entries in the cache holding L1CD descriptors. If this is zero then it is treated as a large number ('infinite') but it is bounded so that the host memory usage of the cache is also bounded.
size_of_l1ste_cache	int	0x0	The number of entries in the cache holding L1STE descriptors. If this is zero then it is treated as a large number ('infinite') but it is bounded so that the host memory usage of the cache is also bounded.
size_of_register_file	int	0x100000	This is the power of two size that the register file occupies in the memory map. It is used to generate a mask for the addresses received on pvbus_control_s to decode the desired register offset. The default for this parameter is 1 MiB.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-444 Parameters for SMMUv3AEM (continued)

Name	Type	Default value	Description
size_of_ste_cache	int	0x0	The number of entries in the cache holding STE structures. If this is zero then it is treated as a large number ('infinite') but it is bounded so that the host memory usage of the cache is also bounded.
size_of_tlb	int	0x0	The number of entries in the TLB. If this is zero then it is treated as a large number ('infinite') but it is bounded so that the host memory usage of the cache is also bounded.
smmu_msi_device_id	int	0x0	When appropriately enabled, assume that MSIs that are generated by the SMMU are presented to the GIC with this DeviceID. See parameter msi_attribute_transform and enable_device_id_checks.
smmuv33_begin_offset_of_qcp0	int	0x0	This is the offset from SMMU_BASE of the first QCP page. The architecture requires that if more than one world of QCPs are present then they are in the order non-secure and then secure QCPs and form one continuous address space in the register file.
support_for_httu_when_starts_disallowed	int	0x0	SMMU_IDR0.HTTU describes to the programmer whether the SMMU and system support HTTU. Typically an SMMU that is capable of HTTU will have a configuration pin that says whether the system supports HTTU or not. The SMMU model determines SMMU_IDR0.HTTU as follows: * If the parameter SMMU_IDR0 indicates any kind of support for HTTU, then the configuration pin turns support on and off between that value and no support for HTTU. * If the parameter SMMU_IDR0 indicates no HTTU support, allow the pin to turn on support to that specified by this parameter. Values for this parameter are the same as for the SMMU_IDR0.HTTU field: * 0 -- no support for HTTU * 1 -- AF flag only * 2 -- AF flag and DBM update
tlb_when_do_f_tlb_conflict_on_overlap	int	0x0	If a TLB entry is created by a walk and it overlaps an existing entry. Then there are some architectural situations where the result is known. For all others, then an implementation is allowed to use an UNPREDICTABLE combination of the two entries, or it can generate F_TLB_CONFLICT: 0 -- never generate 1 -- sometimes generate 2 -- always generate Conflicts between global and non-global entries are not detected by the model.
translated_device_id_base	int	0x0	When appropriately enabled, assume that client device accesses are translated to a DeviceID as seen by the GIC of: StreamID + translated_device_id_base See parameter output_attribute_transform and enable_device_id_checks.
treat_debug_read_accesses_as_speculative_accesses	bool	0x0	The SMMU architecture has the concept of speculative accesses. If you set this flag to true, then debug read accesses flowing from the upstream system through the SMMU will be interpreted as speculative. The difference is that a speculative read will: * participate in HTTU * if it encounters a (non-HTTU) fault will always return abort Debug writes are still considered as debug accesses. All speculative writes would be aborted and this is not a useful behaviour for the SMMU to emulate.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-444 Parameters for SMMUv3AEM (continued)

Name	Type	Default value	Description
tw_qs_attribute_transform	string	""	This transforms the attributes of table walk and queue transactions. * ""/"none" -- no transform * How to alter the output attributes. An example is: "ExtendedID[35:32]=HWATTR_KIND_0 The attributes that can appear on the left hand side of the transform are ExtendedID, MasterID and UserFlags. The source attributes that can be used are: * ExtendedID/MasterID/UserFlags -- the incoming attributes. * HWATTR_KIND_0 -- PBHA information The right hand side may also contain numeric literals. Any bits of the attributes with no transform are unchanged. NOTE that MPAM information comes from the parameter mpam_attribute_transform. See also output_attribute_transform and msi_attribute_transform.
unpred_httu_percent_do_discretionary_AF	int	0x32	If a descriptor could have a discretionary update of the AF flag on then what is the percentage of the time that the AF update should occur.
unpred_httu_percent_do_discretionary_DBM	int	0x32	If a descriptor could have a discretionary DBM update to make the descriptor WriteableDirty then what is the percent of the time time that the DBM update should occur.
unpred_translated_access_out_of_range_of_oas	int	0x1	If a Translated Access is presented to the SMMU that is > OAS then it is CONSTRAINED UNPRED as to whether the transaction will either: 0 -- be truncated to OAS and go downstream 1 -- be aborted, no event written
wait_atos_ticks	int	0x0	This is the time to wait before doing an ATOS operation. If bit 32 is set (0x1_0000_0000) then the time waited for is a uniform randomly distributed time [0,max(2,(t & 0xFFFFffff))).
wait_cmdq_ticks	int	0x0	This is the time to wait before doing something on the command queue. If bit 32 is set (0x1_0000_0000) then the time waited for is a uniform randomly distributed time [0,max(2,(t & 0xFFFFffff))).
wait_eventq_ticks	int	0x0	This is the time to wait before doing something on the event queue. If bit 32 is set (0x1_0000_0000) then the time waited for is a uniform randomly distributed time [0,max(2,(t & 0xFFFFffff))).
wait_msi_ticks	int	0x0	This is the time to wait before sending an MSI. If bit 32 is set (0x1_0000_0000) then the time waited for is a uniform randomly distributed time [0,max(2,(t & 0xFFFFffff))).
wait_pri_req_ticks	int	0x0	This is the time to wait before processing a PRI Request. If bit 32 is set (0x1_0000_0000) then the time waited for is a uniform randomly distributed time [0,max(2,(t & 0xFFFFffff))).
wait_pri_resp_ticks	int	0x1	This is the time to wait before sending a PRI Response back to the PCIe subsystem. When a PRI Response is an auto-response then the ATC might immediately make a new ATS request, that immediately fails, that immediately makes a PRI Request, that auto-responds, etc. To break this loop, then we introduce a minimum time on all PRI Responses to give other components in the system a chance to run. If bit 32 is set (0x1_0000_0000) then the time waited for is a uniform randomly distributed time [0,max(2,(t & 0xFFFFffff))).

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-444 Parameters for SMMUv3AEM (continued)

Name	Type	Default value	Description
`when_fetch_vms`	`int`	`0x0`	Architecturally, there is flexibility in how a VMS is cached and thus: * when it will be fetched * the prioritization of F_VMS_FETCH. Of the many architecturally-allowed options, the model offers two: 0 -- fetched and cached immediately after the STE is fetched 1 -- fetched and cached immediately after the CD is fetched In both cases, then the VMS is cached in the STE and CMD_CFGI_VMS_PIDM is a NOP.
`width_of_agbpa_impdef`	`int`	`0x10`	Width of the SMMU_s_AGBPA.IMPDEF field.

3.9.65 SMSC_91C111

10/100 Non-PCI Ethernet Controller(SMSC 91C111). This model is written in C++.

SMSC_91C111 contains the following CADI targets:

- SMSC_91C111

SMSC_91C111 contains the following MTI components:

- *4.107 PVBusSlave - trace* on page 4-2760

About SMSC_91C111

This component provides the register interface of the SMSC part and can be configured to act as an unconnected Ethernet port, or an Ethernet port connected to the host by an Ethernet bridge.

It uses a banked register model of primarily 16-bit registers. There are also indirectly accessible registers for the PHY unit.

If a MAC address is not specified in the `mac_address` parameter, the simulator takes the default MAC address, which is randomly generated. This provides some degree of MAC address uniqueness when running models on multiple hosts on a local network.

Note

DHCP servers allocate the IP addresses, but because they sometimes do this based on the MAC address provided to them, using random MAC addresses might interact unfortunately with some DHCP servers.

Table 3-445 Ports

Name	Protocol	Type	Description
`clock`	`ClockSignal`	Slave	Clock input, typically 25MHz, which sets the master transmit/receive rate.
`eth`	*`VirtualEthernet` on page 2-79*	Master	Ethernet port.
`intr`	*`Signal` on page 2-85*	Master	Interrupt signal.
`pvbus`	`PVBus`	Slave	Slave port for register access.
`state`	`ValueState_64`	Master	State port to retrieve state of host bridge

Related tasks

Configuring the networking environment for Microsoft Windows on page 1-39

Configuring the networking environment for Linux on page 1-44

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

3.9.66 SP804_Timer

ARM Dual-Timer Module(SP804). This model is written in LISA+.

SP804_Timer contains the following CADI targets:

- ClockDivider
- CounterModule
- SP804_Timer

SP804_Timer contains the following MTI components:

- *4.60 ClockDivider - trace* on page 4-2438
- *4.107 PVBusSlave - trace* on page 4-2760

Table 3-446 Ports

Name	Protocol	Type	Description
clock	ClockSignal	Slave	Clock input, typically 1MHz, driving master count rate.
irq_out0	*Signal on page 2-85*	Master	Interrupt signaling.
irq_out1	*Signal on page 2-85*	Master	Interrupt signaling.
pvbus	PVBus	Slave	Slave port for register access.
timer_en[2]	*ClockRateControl on page 2-65*	Slave	Port for changing the rate of timer n.

3.9.67 SP805_Watchdog

ARM Watchdog Module(SP805). This model is written in LISA+.

SP805_Watchdog contains the following CADI targets:

- SP805_Watchdog

SP805_Watchdog contains the following MTI components:

- *4.107 PVBusSlave - trace* on page 4-2760

Table 3-447 Ports

Name	Protocol	Type	Description
clk_in	ClockSignal	Slave	Clock input, typically 1MHz, driving master count rate.
irq_out	*Signal on page 2-85*	Master	Interrupt signaling.
pvbus_s	PVBus	Slave	Slave port for register access.
reset_in	*Signal on page 2-85*	Slave	Reset signaling.
reset_out	*Signal on page 2-85*	Master	Reset signaling.

Table 3-448 Parameters for SP805_Watchdog

Name	Type	Default value	Description
simhalt	bool	0x0	Halt on reset.

3.9.68 SP810_SysCtrl

PrimeXsys System Controller(SP810) NB: Only EB relevant functionalities are fully implemented. This model is written in LISA+.

SP810_SysCtrl contains the following CADI targets:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

- ClockDivider
- SP810_SysCtrl

SP810_SysCtrl contains the following MTI components:

- *4.60 ClockDivider - trace* on page 4-2438
- *4.107 PVBusSlave - trace* on page 4-2760

Table 3-449 Ports

Name	Protocol	Type	Description
clk_in	ClockSignal	Slave	Clock input.
hclkdivsel	*ValueState* on page 2-85	Master	Define the processor clock/bus clock ratio. Not fully implemented. Using this port has unpredictable results.
npor	*Signal* on page 2-85	Slave	Power on reset. Not fully implemented. Using this port has unpredictable results.
pll_en	*Signal* on page 2-85	Master	PLL enable output. Not fully implemented. Using this port has unpredictable results.
pvbus	PVBus	Slave	Slave port for register access.
ref_clk_in	ClockSignal	Slave	Clock source used by the Timer and Watchdog modules.
remap_clear	*StateSignal* on page 2-85	Master	Remap clear request output.
remap_stat	*StateSignal* on page 2-85	Slave	Remap status input. Not fully implemented. Using this port has unpredictable results.
sleep_mode	*Signal* on page 2-85	Master	Control clocks for SLEEP mode. Not fully implemented. Using this port has unpredictable results.
sys_id	*ValueState* on page 2-85	Slave	Unused port.
sys_mode	*ValueState* on page 2-85	Slave	Present system mode. Not fully implemented. Using this port has unpredictable results.
sys_stat	*ValueState* on page 2-85	Slave	System status input. Not fully implemented. Using this port has unpredictable results.
timer_clk_en[4]	*ClockRateControl* on page 2-65	Master	Timer clock enable n.
wd_clk_en	*Signal* on page 2-85	Master	Watchdog module clock enable output. Not fully implemented. Using this port has unpredictable results.
wd_en	*Signal* on page 2-85	Slave	Watchdog module enable input. Not fully implemented. Using this port has unpredictable results.

Table 3-450 Parameters for SP810_SysCtrl

Name	Type	Default value	Description
sysid	int	0x0	System Identification Register.
use_s8	bool	0x0	Use Switch 8 (S1-S4)

3.9.69 TZC_400

TrustZone Address Space Controller. This model is written in C++.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

TZC_400 contains the following CADI targets:

- TZC_400

TZC_400 contains the following MTI components:

- *4.105 PVBusMapper - trace* on page 4-2758
- *4.107 PVBusSlave - trace* on page 4-2760

About TZC_400

The TZC-400 determines, under software control, whether a particular bus master is permitted to issue Non-secure accesses to a particular physical address.

The component has:

- Eight address regions in addition to the base region, region 0.
- A programmable control block for security-access permissions configuration through the *Advanced Peripheral Bus* (APB).
- Up to four address filters that share common set region set-up registers.
- Software configurable permission check failure reporting and interrupt signaling.
- Filtering with a *Non-Secure Access ID* (NSAID).
- A gate keeper, to allow or block accesses to the filter unit.
- Configurable reset values of region configuration registers and other key configuration registers.

This component has the following subcomponents:

TZFilterUnits
: The TZC-400 has four TZFilterUnits. The `BUILD_CONFIG` register sets the configuration. The `rst_build_config` parameter controls the register. The value of `rst_build_config` varies with the system. See the system design documentation or system integration documentation. For AEMv8-A, it is `0x3003F08`.

TZDummyDevice
: An internal dummy device that mimics RAZ/WI for TZFilterUnits. The system uses it when there is a permission violation and a bus returns Transaction OK.

Note

- Configure `master_id_from_label` or `id_mapping`, `rst_build_config`, and `rst_region_attributes_0` before running the model to set the desired behaviors. Otherwise, the system resets all region configuration registers, `rst_action`, and `rst_gate_keeper` to `0`, and resets `rst_build_config` and `rst_region_attributes_0` to sensible default values.
- Configure either `id_mapping` or `master_id_from_label` at model init, or a warning message appears.
- The syntax of `id_mapping` is:

  ```
  <masterid_0>:<nsaid_0>,<masterid_1>:<nsaid_1>,<masterid_n>:<nsaid_n>
  ```

  Separate the mapping pairs by a comma. The `masterid` is the ID of the bus master, such as the parameter `CLUSTER_ID` on Cortex-A15/7, `cluster_id` port of Cortex-A15/7, or `master_id` parameter for Cortex-M3.

Differences between the model and the RTL

Unlike the hardware, this component does not have:

- Asynchronous clocks. The model does not need clocks for data transfer, or clock signals.
- *QoS Virtual Network* (QVN) support. Specifically, it does not implement the vnet bits[27:24] in `FAIL_ID_<x>` registers.
- Fast Path and Fast Path ID. In the model, transactions occur at similar speeds.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

- 256 outstanding accesses globally for each read or write Normal Paths and configurable 8, 16, or 32 outstanding accesses on Fast Path read access. The model does not support QVN. This concept is meaningless for a PV level model.
- Configurable address bus width, data bus width, transaction ID tag, and USER bus width. A single bus implementation, PVBus, covers these AXI bus hardware implementation details.

This component does not implement:

- The vnet bits[27:24] in `FAIL_ID_<x>` registers.
- Any background logic for the speculation control register. This does not affect model behavior.

Table 3-451 Ports

Name	Protocol	Type	Description
`apbslave_s`	`PVBus`	Slave	Bus access for control register.
`filter_pvbus_m[4]`	`PVBus`	Master	Outgoing bus traffic from filter units.
`filter_pvbus_s[4]`	`PVBus`	Slave	Incoming bus traffic to filter units.
`tzc_reset`	*Signal* on page 2-85	Slave	Reset signal from external master.
`tzcint`	*Signal* on page 2-85	Master	TrustZone interrupt signal, controlled by ACTION register.

3.9.70 TZFilterUnit

TrustZone Filter Unit. This model is written in C++.

TZFilterUnit contains the following CADI targets:

- TZFilterUnit

TZFilterUnit contains the following MTI components:

- *4.105 PVBusMapper - trace* on page 4-2758

Table 3-452 Ports

Name	Protocol	Type	Description
`control`	*TZFilterControl* on page 2-79	Master	Configuration port.
`pvbus_m`	`PVBus`	Master	Master bus port.
`pvbus_s`	`PVBus`	Slave	Slave bus port.

3.9.71 TZIC

ARM TrustZone Interrupt Controller(SP890). This model is written in LISA+.

TZIC contains the following CADI targets:

- TZIC

TZIC contains the following MTI components:

- *4.107 PVBusSlave - trace* on page 4-2760

About TZIC

The TZIC provides a software interface to the secure interrupt system in a TrustZone design. It provides secure control of the nFIQ and masks out the interrupt sources chosen for nFIQ from the interrupts that are passed onto a non-secure interrupt controller.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-453 Ports

Name	Protocol	Type	Description
fiq_out	*Signal* on page 2-85	Master	FIQ interrupt to processor.
input[32]	*Signal* on page 2-85	Slave	32 interrupt input sources.
irq_out[32]	*Signal* on page 2-85	Master	32 IRQ output ports.
nsfiq_in	*Signal* on page 2-85	Slave	Connects to the nFIQ output of the non-secure interrupt controller.
pvbus	PVBus	Slave	Slave port for connection to PV bus master/decoder.
sfiq_in	*Signal* on page 2-85	Slave	Daisy chaining secure FIQ input, otherwise connects to logic 1 if interrupt controller not daisy chained.

3.9.72 v7_VGIC

System VGIC architecture version v7. This model is written in C++.

v7_VGIC contains the following CADI targets:

- v7_VGIC

v7_VGIC contains the following MTI components:

- *4.107 PVBusSlave - trace* on page 4-2760
- *4.124 v7_VGIC - trace* on page 4-2886

Table 3-454 Ports

Name	Protocol	Type	Description
cfgsdisable	*Signal* on page 2-85	Slave	Disable write access to some GIC registers.
configuration	v7_VGIC_Configuration_Protocol	Slave	Configure the mapping of the core number (from MasterID) to the core interface number.
fiq_in[8]	*Signal* on page 2-85	Slave	FIQ inputs.
fiq_out[8]	*Signal* on page 2-85	Master	FIQ outputs.
irq_in[8]	*Signal* on page 2-85	Slave	IRQ intputs.
irq_out[8]	*Signal* on page 2-85	Master	IRQ outputs.
ppi_core0[16]	*Signal* on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 0.
ppi_core1[16]	*Signal* on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 1.
ppi_core2[16]	*Signal* on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 2.
ppi_core3[16]	*Signal* on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 3.
ppi_core4[16]	*Signal* on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 4.
ppi_core5[16]	*Signal* on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 5.
ppi_core6[16]	*Signal* on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 6.
ppi_core7[16]	*Signal* on page 2-85	Slave	Private peripheral interrupts (ID16-ID31) for cpu 7.
pvbus_s	PVBus	Slave	Bus port for accessing distributor registers.
reporting_interface	VGICReportingProtocol	Slave	Logging interface.
reset_signal	*Signal* on page 2-85	Slave	Reset signal input.
spi[988]	*Signal* on page 2-85	Slave	SPI inputs.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 3-454 Ports (continued)

Name	Protocol	Type	Description
vfiq_out[8]	*Signal* on page 2-85	Master	Virtual FIQ outputs.
virq_out[8]	*Signal* on page 2-85	Master	Virtual IRQ outputs.
wakeup_fiq[8]	*Signal* on page 2-85	Master	Wakeup signal for FIQ.
wakeup_irq[8]	*Signal* on page 2-85	Master	Wakeup signal for IRQ.

Table 3-455 Parameters for v7_VGIC

Name	Type	Default value	Description
enable_log_errors	bool	0x0	Enable logging of errors
enable_log_fatal	bool	0x0	Enable logging of fatal errors
enable_log_warnings	bool	0x0	Enable logging of warnings
enabled	bool	0x1	Enable the component. If it is disabled then all register writes will have no effect.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Chapter 4
Trace components

This chapter describes the trace components that are provided by Fast Models components.

It contains the following sections:

- *4.1 AHCI_SATA - trace* on page 4-864.
- *4.2 AMBAPVSignal2SGSignalx8 - trace* on page 4-869.
- *4.3 ARM_AEM-A_MP - trace* on page 4-870.
- *4.4 ARM_AEMv8-A_MP - trace* on page 4-908.
- *4.5 ARM_AEMv8M - trace* on page 4-946.
- *4.6 ARM_Cortex-A15 - trace* on page 4-979.
- *4.7 ARM_Cortex-A17 - trace* on page 4-1015.
- *4.8 ARM_Cortex-A32 - trace* on page 4-1051.
- *4.9 ARM_Cortex-A35 - trace* on page 4-1089.
- *4.10 ARM_Cortex-A5 - trace* on page 4-1127.
- *4.11 ARM_Cortex-A53 - trace* on page 4-1152.
- *4.12 ARM_Cortex-A55 - trace* on page 4-1190.
- *4.13 ARM_Cortex-A57 - trace* on page 4-1228.
- *4.14 ARM_Cortex-A5MP - trace* on page 4-1266.
- *4.15 ARM_Cortex-A65 - trace* on page 4-1291.
- *4.16 ARM_Cortex-A65AE - trace* on page 4-1329.
- *4.17 ARM_Cortex-A7 - trace* on page 4-1367.
- *4.18 ARM_Cortex-A72 - trace* on page 4-1403.
- *4.19 ARM_Cortex-A73 - trace* on page 4-1441.
- *4.20 ARM_Cortex-A75 - trace* on page 4-1479.
- *4.21 ARM_Cortex-A76 - trace* on page 4-1517.
- *4.22 ARM_Cortex-A76AE - trace* on page 4-1556.
- *4.23 ARM_Cortex-A77 - trace* on page 4-1595.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

- *4.24 ARM_Cortex-A78 - trace* on page 4-1634.
- *4.25 ARM_Cortex-A78C - trace* on page 4-1673.
- *4.26 ARM_Cortex-A8 - trace* on page 4-1712.
- *4.27 ARM_Cortex-A9MP - trace* on page 4-1736.
- *4.28 ARM_Cortex-A9UP - trace* on page 4-1772.
- *4.29 ARM_Cortex-M0 - trace* on page 4-1808.
- *4.30 ARM_Cortex-M0+ - trace* on page 4-1839.
- *4.31 ARM_Cortex-M23 - trace* on page 4-1870.
- *4.32 ARM_Cortex-M3 - trace* on page 4-1901.
- *4.33 ARM_Cortex-M33 - trace* on page 4-1921.
- *4.34 ARM_Cortex-M35P - trace* on page 4-1954.
- *4.35 ARM_Cortex-M4 - trace* on page 4-1987.
- *4.36 ARM_Cortex-M55 - trace* on page 4-2009.
- *4.37 ARM_Cortex-M7 - trace* on page 4-2047.
- *4.38 ARM_Cortex-R4 - trace* on page 4-2080.
- *4.39 ARM_Cortex-R5 - trace* on page 4-2103.
- *4.40 ARM_Cortex-R7 - trace* on page 4-2126.
- *4.41 ARM_Cortex-R8 - trace* on page 4-2161.
- *4.42 ARM_Cortex-X1 - trace* on page 4-2196.
- *4.43 ARM_CortexR52 - trace* on page 4-2235.
- *4.44 ARM_Neoverse-E1 - trace* on page 4-2273.
- *4.45 ARM_Neoverse-N1 - trace* on page 4-2311.
- *4.46 ARM_SC000 - trace* on page 4-2350.
- *4.47 ARM_SC300 - trace* on page 4-2381.
- *4.48 ARMv8Cluster - trace* on page 4-2401.
- *4.49 AsyncCacheFlushUnit - trace* on page 4-2402.
- *4.50 Base_PowerController - trace* on page 4-2403.
- *4.51 CCI400 - trace* on page 4-2405.
- *4.52 CCI500 - trace* on page 4-2407.
- *4.53 CCI550 - trace* on page 4-2412.
- *4.54 CCNCache - trace* on page 4-2417.
- *4.55 CCNRegisterSet - trace* on page 4-2422.
- *4.56 CMN600 - trace* on page 4-2424.
- *4.57 CMN600CML - trace* on page 4-2428.
- *4.58 CMN600Cache - trace* on page 4-2429.
- *4.59 CMN650 - trace* on page 4-2434.
- *4.60 ClockDivider - trace* on page 4-2438.
- *4.61 D71 - trace* on page 4-2439.
- *4.62 DMC-500 - trace* on page 4-2448.
- *4.63 DMC-520 - trace* on page 4-2450.
- *4.64 DMC-620 - trace* on page 4-2452.
- *4.65 DP500 - trace* on page 4-2455.
- *4.66 DP550 - trace* on page 4-2457.
- *4.67 DP650 - trace* on page 4-2460.
- *4.68 DSU - trace* on page 4-2463.
- *4.69 DualClusterSystemConfigurationBlock - trace* on page 4-2464.
- *4.70 GIC_400 - trace* on page 4-2466.
- *4.71 GICv2 - trace* on page 4-2470.
- *4.72 GICv3CPUInterface - trace* on page 4-2474.
- *4.73 GICv3CPUInterfaceDecoder - trace* on page 4-2489.
- *4.74 GICv3CommsLogger - trace* on page 4-2490.
- *4.75 GICv3CommsPVBus - trace* on page 4-2494.
- *4.76 GICv3Distributor - trace* on page 4-2496.
- *4.77 GICv3IRI - trace* on page 4-2504.
- *4.78 GICv3InterruptTranslationService - trace* on page 4-2505.
- *4.79 GICv3ProtocolChecker - trace* on page 4-2526.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

- *4.80 GICv3Redistributor - trace* on page 4-2527.
- *4.81 GICv3RedistributorInternal - trace* on page 4-2541.
- *4.82 MMU_400_BASE - trace* on page 4-2547.
- *4.83 MMU_500_BASE - trace* on page 4-2576.
- *4.84 MMU_600 - trace* on page 4-2605.
- *4.85 MMU_700 - trace* on page 4-2664.
- *4.86 MSIRewriter - trace* on page 4-2723.
- *4.87 Mali_G51 - trace* on page 4-2724.
- *4.88 Mali_G71 - trace* on page 4-2726.
- *4.89 Mali_G72 - trace* on page 4-2728.
- *4.90 Mali_G76 - trace* on page 4-2730.
- *4.91 MemoryMappedCounterModule - trace* on page 4-2732.
- *4.92 MessageHandlingUnitV2 - trace* on page 4-2733.
- *4.93 NonVolatileCounter - trace* on page 4-2736.
- *4.94 PL011_Uart - trace* on page 4-2737.
- *4.95 PL11x_CLCD - trace* on page 4-2738.
- *4.96 PL310_L2CC - trace* on page 4-2739.
- *4.97 PL330_DMAC - trace* on page 4-2742.
- *4.98 PPUv0 - trace* on page 4-2747.
- *4.99 PPUv1 - trace* on page 4-2748.
- *4.100 PVBus2AMBAPVACE - trace* on page 4-2750.
- *4.101 PVBusBridge - trace* on page 4-2751.
- *4.102 PVBusExclusiveMonitor - trace* on page 4-2752.
- *4.103 PVBusGICv3Comms - trace* on page 4-2754.
- *4.104 PVBusLogger - trace* on page 4-2756.
- *4.105 PVBusMapper - trace* on page 4-2758.
- *4.106 PVBusMaster - trace* on page 4-2759.
- *4.107 PVBusSlave - trace* on page 4-2760.
- *4.108 PVCache - trace* on page 4-2761.
- *4.109 PVWriteBuffer - trace* on page 4-2767.
- *4.110 RAMDevice - trace* on page 4-2769.
- *4.111 SMMUv3AEM - trace* on page 4-2770.
- *4.112 SMMUv3TestEngine - trace* on page 4-2827.
- *4.113 SignalLogger - trace* on page 4-2833.
- *4.114 TLB - trace* on page 4-2834.
- *4.115 V550 - trace* on page 4-2838.
- *4.116 V61 - trace* on page 4-2852.
- *4.117 Value64Logger - trace* on page 4-2866.
- *4.118 ValueLogger - trace* on page 4-2867.
- *4.119 VirtioBlockDevice - trace* on page 4-2868.
- *4.120 VirtioP9Device - trace* on page 4-2871.
- *4.121 VirtioPCIBlockDevice - trace* on page 4-2872.
- *4.122 VirtualEthernetCrossover - trace* on page 4-2875.
- *4.123 atc - trace* on page 4-2876.
- *4.124 v7_VGIC - trace* on page 4-2886.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.1 AHCI_SATA - trace

This section describes the trace sources.

AHCI_REG_GHC_READ

GHC Register read access to AHCI. Fields:

ADDRESS unsigned int
: Address Offset within GHC space

DATA unsigned int
: Data returned by this register

GHC_REGISTER enum
: Name of the GHC register

AHCI_REG_GHC_WRITE

GHC Register write access to AHCI. Fields:

ADDRESS unsigned int
: Address Offset within GHC space

DATA unsigned int
: Data written to this register

GHC_REGISTER enum
: Name of the GHC register

AHCI_REG_PORT_READ

PORT Register read access to AHCI. Fields:

ADDRESS unsigned int
: Address Offset within PORTx space

DATA unsigned int
: Data returned by this register

PORT unsigned int
: SATA port number

PORTx_REGISTER enum
: Name of the PORTx register

AHCI_REG_PORT_WRITE

PORT Register write access to AHCI. Fields:

ADDRESS unsigned int
: Address Offset within PORTx space

DATA unsigned int
: Data written to this register

PORT unsigned int
: SATA port number

PORTx_REGISTER enum
: Name of the PORTx register

ATA_COMMAND

ATA command received by SATA device. Fields:

COMMAND enum
: ATA Command

FIS_COUNT unsigned int
: Raw value of the Count field in the FIS. (Sector count for non-NCQ commands.)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

FIS_DEVICE unsigned int
: Raw value of the Device fiels on the FIS. Bit 6 indicates LBA (28 or 48 bit access).

FIS_FEATURE unsigned int
: Raw value of the Feature field in the FIS. (Sector count for NCQ commands.)

LBA unsigned int
: Starting sector index

PORT unsigned int
: SATA port number (0-31)

SECTOR_COUNT unsigned int
: Sector count

ATA_COMMAND_HEADER

ATA Command header. Fields:

CFIS_LEN unsigned int
: CFIS length

CMD_TABLE_BASE unsigned int
: Command table Base

IS_WRITE bool
: Write?

PORT unsigned int
: SATA port number (0-31)

PRD_COUNT unsigned int
: Number of PRDs

PRD_TABLE_LEN unsigned int
: PRD table length

ArchMsg.Error.invalid_access_width

Access with invalid access width in AHCI. Fields:

ACCESS_WIDTH unsigned int
: Access Width to a register

ADDRESS unsigned int
: Address Offset within GHC space

DATA unsigned int
: Data with access width of 32 bits

ArchMsg.Error.invalid_register

Access to invalid register offsets in AHCI. Fields:

ADDRESS unsigned int
: Address Offset within GHC or port space

PORT unsigned int
: SATA port number or 0xff for GHC

CFIS

Command FIS. Fields:

CFIS unsigned int
: CFIS content

PORT unsigned int
: SATA port number (0-31)

CMD_SLOT

Command Slot in Command List. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CMD_SLOT unsigned int
: Command Slot

PORT unsigned int
: SATA port number (0-31)

DATA_READ

Data read from ahci_dma_m. Fields:

ADDRESS unsigned int
: Address

DATA unsigned int
: Data Read

PORT unsigned int
: SATA port number (0-31)

DATA_READ_ABORT

The data read by the device aborted. Fields:

ADDR unsigned int
: Address.

NS bool
: Non-secure access.

PORT unsigned int
: SATA port number (0-31)

SIZE unsigned int
: The data size in bytes.

DATA_WRITE

Data write from ahci_dma_m. Fields:

ADDRESS unsigned int
: Address

DATA unsigned int
: Data Write

PORT unsigned int
: SATA port number (0-31)

DATA_WRITE_ABORT

The data write by the device aborted. Fields:

ADDR unsigned int
: Address.

NS bool
: Non-secure access.

PORT unsigned int
: SATA port number (0-31)

SIZE unsigned int
: The data size in bytes.

FIS

FIS received by AHCI controller from SATA device. Fields:

DATA unsigned int
: FIS content (first byte is the FIS type)

PORT unsigned int
: SATA port number (0-31)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MSI_INTERRUPT

MSI interrupt generated by AHCI. Fields:

GHC_IS unsigned int
: Contents of GHC.IS register

MSI_VECTOR unsigned int
: MSI vector number

PORT_INTRRUPT

Interrupt generated by port. This is emitted only when the corresponding interrupt is enabled in PxIE, but regardless whether interrupts are globally enabled in GHC.IE. Fields:

INTRRUPT_TYPE enum
: Interrupt type

PORT unsigned int
: SATA port number (0-31)

PRDT_ENTRY

PRDT_ENTRY. Fields:

ADDR unsigned int
: PRD entry address

COUNT unsigned int
: Bytes for this PRD entry

IRQ bool
: Raise an interrupt upon completion

PORT unsigned int
: SATA port number (0-31)

SATA_BS_PERF

SATA backend storage periodic performance indicator. This allows to monitor the data throughput in terms of host time. This emits a trace event, at most once per second, which indicates the data rate for reads, writes and the sum (througput) in the interval since the last event. An event is only generated when there was at least one read/write access in the last second. Fields:

PERF string
: Throughput/read/write performance as a formatted string.

SATA_PORT unsigned int
: Index of the SATA port this backend storage is attached to.

SATA_BS_READ

SATA backend storage read. Fields:

SATA_PORT unsigned int
: Index of the SATA port this backend storage is attached to.

SECTOR_COUNT unsigned int
: Number of sectors read (512 byte sectors)

SECTOR_INDEX unsigned int
: Access starts at this sector index (512 byte sectors)

SATA_BS_READ_DATA

SATA backend storage read. This trace is issued for each individual 512-byte sector of the original access size. Fields:

DATA unsigned int
: Sector Data (always 512 bytes)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SATA_PORT unsigned int
: Index of the SATA port this backend storage is attached to.

SECTOR_INDEX unsigned int
: Sector index of this data (512 byte sectors)

SATA_BS_TRIM

SATA backend storage trim (marking sectors as unused). Fields:

SATA_PORT unsigned int
: Index of the SATA port this backend storage is attached to.

SECTOR_COUNT unsigned int
: Number of sectors read (512 byte sectors)

SECTOR_INDEX unsigned int
: Access starts at this sector index (512 byte sectors)

SATA_BS_WRITE

SATA backend storage write. Fields:

SATA_PORT unsigned int
: Index of the SATA port this backend storage is attached to.

SECTOR_COUNT unsigned int
: Number of sectors read (512 byte sectors)

SECTOR_INDEX unsigned int
: Access starts at this sector index (512 byte sectors)

SATA_BS_WRITE_DATA

SATA backend storage write. This trace is issued for each individual 512-byte sector of the original access size. Fields:

DATA unsigned int
: Sector Data (always 512 bytes)

SATA_PORT unsigned int
: Index of the SATA port this backend storage is attached to.

SECTOR_INDEX unsigned int
: Sector index of this data (512 byte sectors)

SET_FEATURE

SET FEATURE. Fields:

PARAMETER unsigned int
: Sub command parameter

PORT unsigned int
: SATA port number (0-31)

SUB_COMMAND enum
: Set Feature sub command

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.2 AMBAPVSignal2SGSignalx8 - trace

This section describes the trace sources.

ArchMsg.Warning.AMBAPVSignal2SGSignal_unconnected_warning

Unconnected Warning. Fields:

export_id `signed int`
: Received export_id

index `unsigned int`
: The index of unconnected toggled signal

state `bool`
: The toggled to Signal state

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.3 ARM_AEM-A_MP - trace

This section describes the trace sources.

ASYNC_MEMORY_FAULT

Context ID Register write. Fields:

FAULT unsigned int
: Fault status in ESR format

PADDR unsigned int
: Physical Address (or 0 if unavailable)

VADDR signed int
: Virtual Address (or 0 if unavailable)

ATOMIC_END_ACCESS

Bus trace access for atomics. Fields:

ACCESS_FAIL bool
: Memory access failed

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

LOAD_VALUE unsigned int
: Loaded values

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ATOMIC_START_ACCESS

Bus trace access for atomics. Fields:

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

NSDESC unsigned int
: The security state of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ArchMsg.Error.exit_code

Fields:

EXITCODE signed int
: Exit code to be returned

KIND string
: Component kind that invoked the exit code trace

REASON string
: Reason why the exit code trace was invoked (optional)

ArchMsg.Warning.IgnoredWarmResetRequest

Event fires when software attempts to request warm reset through write to deprecated DBGPRCR.CWRR.

ArchMsg.Warning.TarmacReconstructor

Unexpected event happened during Tarmac reconstruction. Fields:

REASON string
: Reason

ArchMsg.Warning.Unpredictably Indexed PM Event Register

Fields:

IsDirect unsigned int
: Direct Access rather than indirect

N unsigned int
: Number of Accessible Registers

SEL unsigned int
: Selector

ArchMsg.Warning.branch_to_unaligned_address

Fields:

ADDR unsigned int
: Unaligned branch address

ArchMsg.Warning.cache_contents_unknown

A cache has been activated without being invalidated and may contain UNKNOWN content. Fields:

REGIME unsigned int
: Current translation regime

SIDE unsigned int
: 0 d-side cache or unified cache, 1 i-side

ArchMsg.Warning.dcimvac_matches_watchpoint

Watchpoints matching an AArch32 DCIMVA are implementation defined.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_fieldmismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_fracbitsnegative

VCVT Instruction is unpredictable with a negative number of fraction bits.

ArchMsg.Warning.decode_initblock

Instruction is unpredictable in an IT block.

ArchMsg.Warning.decode_initblocknotlast

Instruction is unpredictable in an IT block when not the last.

ArchMsg.Warning.decode_invalid_condition

Instruction is unpredictable when the condition is not AL.

ArchMsg.Warning.decode_invalidfieldcombination

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_invalidvalue

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_mem_hint_unallocated

There is no memory hint allocated to this bit pattern.

ArchMsg.Warning.decode_registermatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registermismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registeroutofrange

Float point instruction is unpredictable indexing beyond the end of the register bank.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_sbzsbo

Reserved bits in the instruction are not canonical.

ArchMsg.Warning.decode_transactiontoobig

Attempt to perform a memory transaction of more than 128 bytes.

ArchMsg.Warning.decode_unpred_other

Instruction is unpredictable.

ArchMsg.Warning.decode_unpreduseofpc

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13inreglist

Instruction is unpredictable with R13 in register-list.

ArchMsg.Warning.decode_useofd16orhigher

Attempt to use register D16 or higher where it is not allowed.

ArchMsg.Warning.decode_writebackandbaseinlist

Instruction is unpredictable with write back when the base register is in the transfer list.

ArchMsg.Warning.decode_zeroregistersinlist

Instruction is unpredictable when no registers are to be transfered.

ArchMsg.Warning.recursive_branch

An instruction is performing a branch that targets the same instruction. If this was intended then a WFI might be more effective. Fields:

PC unsigned int
: Address of instruction causing the branch

ArchMsg.Warning.recursive_exception

An instruction has generated an exception that targets the same instruction. Fields:

CPSR unsigned int
: Processor state

PC unsigned int
: Address of instruction causing the exception

TYPE enum
: Type of exception

ArchMsg.Warning.reentrant_vector_catch

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR unsigned int
vector address

ArchMsg.Warning.reserved_it_state

Fields:

ITSTATE unsigned int
reserved IT state

ArchMsg.Warning.tlb_contents_unknown

Translation has been enabled without invalidating the TLB. Fields:

EL unsigned int
Exception level of translation regime
INVALIDITY unsigned int
Which TLB may be invalid (if TLBs are separate)
NON_SECURE unsigned int
Security state of translation regime
VMID unsigned int
Current VMID, if applicable

ArchMsg.Warning.unknown_DLR_DSPSR

Exiting debug state with unknown DLR or DSPSR.

ArchMsg.Warning.unknown_ELR_SPSR

Returning from debug exception with unknown ELR or SPSR.

ArchMsg.Warning.unpredictable_a32_breakpoint

Fields:

ADDR1 unsigned int
Address
ADDR2 unsigned int
Breakpoint address
BAS unsigned int
BAS Field
IS_ADDRESS_MISMATCH_BREAKPOINT bool
Is Mismatch breakpoint, not a match breakpoint
N unsigned int
Breakpoint number
OFFSET unsigned int
Indicates which halfword actually considering into the A32 instruction

ArchMsg.Warning.unpredictable_banked_register_access

Fields:

R unsigned int
R bit
SYSM unsigned int
SYSm

ArchMsg.Warning.unpredictable_ccfail_undef

Fields:

OPCODE unsigned int
opcode of the instruction

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VADDR unsigned int
Virtual address of the instruction being executed

ArchMsg.Warning.unpredictable_ici_bits

ICI bits do not apply or are architecturally unpredictable for instruction.

ArchMsg.Warning.unpredictable_t32_breakpoint

Fields:

ADDR1 unsigned int
Address
ADDR2 unsigned int
Breakpoint address
BAS unsigned int
BAS Field
IS_ADDRESS_MISMATCH_BREAKPOINT bool
Is Mismatch breakpoint, not a match breakpoint
N unsigned int
Breakpoint number
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_t32_vector_catch

Fields:

ADDR unsigned int
Address
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_watchpoint_far

Fields:

LOWER_BOUND unsigned int
the lowest address accessed by the instruction that triggered the watchpoint
UPPER_BOUND unsigned int
the highest watchpointed address accessed by that instruction

ArchMsg.Warning.warning_AdvSIMDExpandImmUnexpectedZero

AdvSIMDExpandImm may treat this immediate value as UNPREDICTABLE.

ArchMsg.Warning.warning_ConditionalSMC

SMC instruction has UNPREDICTABLE effects when conditional when combined with traps.

ArchMsg.Warning.warning_access_crosses_page_boundary

Access crosses page boundary.

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault_on_both_pages

Access crosses page boundary has fault(s).

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_access_crosses_page_boundary_spanning_different_memory

Fields:

ADDR unsigned int
: address of access crossing page

MEMTYPE_PAGE1 signed int
: First Page's memory type (0-Normal, 1-Non-Normal)

MEMTYPE_PAGE2 signed int
: Second Page's memory type (0-Normal, 1-Non-Normal)

ArchMsg.Warning.warning_access_wraps_around_memory

Fields:

WIDTH unsigned int
: Address width

ArchMsg.Warning.warning_bx_from_thumbee

Fields:

ADDR unsigned int
: Destination address

ArchMsg.Warning.warning_ccsidr_unimplemented_level

Fields:

CSSELR unsigned int
: current effective value of CSSELR

ArchMsg.Warning.warning_change_to_ns_when_tge_set

Attempting to change NS to 1 when HCR.TGE = 1.

ArchMsg.Warning.warning_contiguous_bit_check_abort

Fields:

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ArchMsg.Warning.warning_contiguous_bit_error

Fields:

CONTIG_ADDR unsigned int
: address of the TLB first read, with which this entry is expected to be contiguous

CONTIG_DATA unsigned int
: expected contents based on the entry at CONTIG_ADDR

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ENTRY_DATA unsigned int
: contents of conflicting TLB entry

ArchMsg.Warning.warning_cp10_cp11_mismatch

Fields:

EL unsigned int
: EL owning the control

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_cp10_cp11_reserved_value

Fields:

CP10 unsigned int
: CP10 access

CP11 unsigned int
: CP11 access

ArchMsg.Warning.warning_csselr_level_out_of_range

Fields:

IMPLEMENTED unsigned int
: number of cache levels implemented

LEVEL unsigned int
: written value of CSSELR.Level

TYPE enum
: The type of cache selected by CSSELR.InD and TnD

ArchMsg.Warning.warning_debug_flow_control_bits_not_obeyed

Fields:

ERROR enum
: type of error resulting

REGISTER enum
: register under consideration

ArchMsg.Warning.warning_debug_register_access_during_reset

Fields:

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

ArchMsg.Warning.warning_decode_cps_inconsistent_fields

The mode setting fields in the CPS instruction are inconsistent.

ArchMsg.Warning.warning_decode_invalid_state

Instruction is unpredictable in current exception-level/security state.

ArchMsg.Warning.warning_default_cacheable_mmu_on

Default cacheable is enabled (HCR.DC = 1) while MMU is enabled.

ArchMsg.Warning.warning_default_cacheable_vmmu_off

Default cacheable is enabled (HCR.DC = 1) while HCR.VM = 0.

ArchMsg.Warning.warning_deprecated_wvr_bit_2_set

Fields:

N unsigned int
: index

WVR unsigned int
: DBGWVR

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_eret_while_software_step_active_pending

Missing ISB between setting MDSCR_EL1.SS and ERET.

ArchMsg.Warning.warning_exclusive_to_non_normal

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_exclusive_to_non_writeback

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_execute_from_device_memory

An attempt was made to execute from Device memory.

ArchMsg.Warning.warning_illegal_cpsr_mode

Fields:

NEW_MODE unsigned int
: New value of CPSR.M

ArchMsg.Warning.warning_illegal_srs_mode

Fields:

MODE unsigned int
: mode for Banked SP

ArchMsg.Warning.warning_implementation_defined_read_debug_register_in_SCS

Read Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_implementation_defined_sequential_security_transitions_supported

The behaviour of a sequential instruction fetches that cross from non-secure to secure memory and contain SG instruction is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_implementation_defined_stack_limit_check_supported

It is IMPLEMENTATION DEFINED whether stack pointer limit checking is performed for the this instructions. Fields:

ADDRESS unsigned int
: Address of instruction

OPCODE unsigned int
: instruction opcode

ArchMsg.Warning.warning_implementation_defined_write_debug_register_in_SCS

Write Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

DATA unsigned int
: data attempted to be written

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_invalid_tcr_granule

Fields:

REQUEST signed int
: page size requested (or 0 for reserved)

SUBSTITUTE signed int
: best guess available page size

TG_ID bool
: bits TG0 or TG1

ArchMsg.Warning.warning_load_multiple_user_registers_from_user_mode

An LDM(user registers) instruction executed from user mode.

ArchMsg.Warning.warning_load_pc_from_unaligned

PC loaded from an unaligned location.

ArchMsg.Warning.warning_mte_in_mem_attr_not_supported

Fields:

ATTR unsigned int
: Memory attribute

ArchMsg.Warning.warning_reserved_breakpoint_state_match

Fields:

IS_BREAKPOINT unsigned int
: Is breakpoint, not watchpoint

ArchMsg.Warning.warning_shareability

Unpredictable: combination of the 1st and 2nd stages of translation is Normal Inner Non-Cacheable, Outer Non-Cacheable.

ArchMsg.Warning.warning_software_step_set_while_enabled

MDSCR_EL1.SS set to 1 while software step debug exceptions are enabled.

ArchMsg.Warning.warning_thumb_instruction_wraps_around_memory

Thumb instruction warps around memory.

ArchMsg.Warning.warning_ttbr_sbz_bits_are_not_zero

Fields:

MSB unsigned int
: x-1

TTBR enum
: which TTBR register

ArchMsg.Warning.warning_unaligned_address_dbgdtrrx_write

Fields:

ADDR unsigned int
: Address

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unaligned_address_dbgdtrtx_read

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_unaligned_to_device

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_unaligned_to_strongly_ordered

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_unknown_sau_rnr

SAU_RNR was set to an unsupported value.

ArchMsg.Warning.warning_unpredictable_AIRCR_PRIP_and_BFHFNMINP

The effect of setting both AIRCR.BFHFNMINP and AIRCR.PRIP to 1 is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTCLRACTIVE_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTCLRACTIVE if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_and_SYSRESETREQ

When the processor is halted in Debug state, if a write to the register writes a 1 to both VECTRESET and SYSRESETREQ, the behavior is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTRESET if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_incorrect_VKEY

The value 0x05FA must be written to AIRCR.VKEY, otherwise the register write is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_EXE_RETURN_Reserved_Bit

EXC_RETURN[23:7] are reserved with the special condition that all bits should be written as one. Values other than all 1s are UNPREDICTABLE. Fields:

EXC_RETURN unsigned int
: Special PC value

SBO_MASK unsigned int
: Which bits Should Be One to not UNPRED

ArchMsg.Warning.warning_unpredictable_change_to_DEMCR_MON_STEP_at_insufficient_priority

The effect of changing DEMCR.MON_STEP at an execution priority that is lower than the priority of the DebugMonitor exception is CONSTRAINED UNPREDICTABLE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_change_to_priority_of_active_exception

Changing the priority of an active exception is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_clear_lspact

Fields:

DRVEXC_TAKEN unsigned int
: derived exception taken not pended

REACHED_LAST_STORE unsigned int
: only the last faulted

ArchMsg.Warning.warning_unpredictable_dbg_exit_unaligned_dlr

Debug exit to AArch32 with DLR[0] set to 1. PC[0] can be set to 0 or to DLR[0].

ArchMsg.Warning.warning_unpredictable_exception_catch

Generation of an exception catch event is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_inconsistent_state

state on exception return is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_instruction

exception return instruction is unpredictable.

ArchMsg.Warning.warning_unpredictable_in_debug_state

Instruction is UNPREDICTABLE when executed in debug state.

ArchMsg.Warning.warning_unpredictable_mair_encoding

Fields:

ATTR unsigned int
: Memory attribute

ArchMsg.Warning.warning_unpredictable_pmu_counter_access

Access to PMU counters from non-secure EL0 or EL1 is UNPREDICTABLE with MDCR_EL2.HPMN set to 0.

ArchMsg.Warning.warning_unpredictable_prioritization_breakpoint_match_vector_catch

Two events with the same priority occurred at the same instruction: Address Matching Vector Catch debug event, and Breakpoint debug event It is constrained unpredictable which is taken.

ArchMsg.Warning.warning_unpredictable_stack_selection

UNPREDICTABLE state on context switch.

ArchMsg.Warning.warning_unpredictable_tsize_out_of_range

Fields:

TCR enum
: Which TCR

TSIZE signed int
: Value of TSize

TSIZE_MAX signed int
: The maximum allowed value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

TTBR unsigned int
: Which TTBR region

ArchMsg.Warning.warning_unpredictable_unaligned_pc_as_base_register

The PC value must be word-aligned, otherwise the behavior of the instruction is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_unaligned_pop_stack

Fields:

FRAME_ADDR unsigned int
: unaligned stack frame address

ArchMsg.Warning.warning_unpredictable_vmsa_memattrib

Fields:

TEXCB unsigned int
: texcb value

TEX_REMAP bool
: use tex remap

ArchMsg.Warning.warning_unpredictable_vtcr_t0sz_sl0

Fields:

SL0 signed int
: VTCR.SL0, starting level for stage 2 translation table walks

T0SZ signed int
: VTCR.T0SZ, required input address range

ArchMsg.Warning.warning_unpredictable_write_DHCSR

Fields:

NEW unsigned int
: data attempted to be written

OLD unsigned int
: previous value

ArchMsg.Warning.warning_unpredictable_write_SHCSR

UNPREDICTABLE setting a pending bit to 1 for an exception with priority >= execution priority.

ArchMsg.Warning.warning_unsupported_access_to_memory_mapped_register

Access to memory mapped register is unsupported.

ArchMsg.Warning.warning_user_jmcr_access

Fields:

WRITE unsigned int
: access is write

ArchMsg.Warning.warning_wcr_mask_and_bas

DBGWCR_EL1.MASK is non-zero and BAS is not 0xff.

ArchMsg.Warning.warning_wcr_mask_reserved

DBGWCR_ELn.MASK is set to a reserved value.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_wcr_non_configuous_bas

DBGWCR_EL1.BAS has non-contiguous set of ones.

ArchMsg.Warning.warning_wvr_masked_bit_not_zero

DBGWVRn_EL1.VA contains a bit masked by DBGWCRn_EL1.MASK that is not zero.

ArchMsg.Why.why_branch_target_exception

A branch target exception has occurred. Fields:

BTYPE unsigned int
: pstate.BTYPE at time branch taken

OPCODE unsigned int
: opcode at branch target

PC unsigned int
: Virtual Address of branch target

ArchMsg.Why.why_illegal_state

An illegal mode change has occured. Fields:

MESSAGE string
: The message

BRANCH_MISPREDICT

Simulating branch mispredict. Fields:

PC unsigned int
: Origin address (or 0 if unavailable)

BRA_DIR

Direct branches, to immediate address. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR

Indirect branches, perhaps to a register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR_CCFAIL

Indirect branches fail, perhaps to a register.

BRA_RET_CCFAIL

Branche return fail, perhaps to a register.

CACHE_MAINTENANCE_OP

Cache Maintenance Operation. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Specified MVA or Set/way

FUNCTION enum
: Clean / Invalidate

SCOPE enum
: Affected region

SIDE enum
: Inst / Data.

CCFAIL

Conditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCFAIL_UNC

Unconditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS

Conditional instruction condition check pass. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS_UNC

Unconditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CHECKPOINT_MESSAGE

Report error messages from the checkpointing process. Fields:

message string
: Message contents.

CHECKPOINT_RESTORE_END

Checkpoint restore completed.

CHECKPOINT_RESTORE_START

Checkpoint restore about to start.

CHECKPOINT_SAVE_END

Checkpoint save completed.

CHECKPOINT_SAVE_START

Checkpoint save about to start.

CODE_CACHE_MAINT

Code cache maintenance. Fields:

END_INCL unsigned int
: Inclusive end addres for by VA or PA requests

NS bool
: Non-secure world for PA

START unsigned int
: Start address for by VA or PA requests

TYPE enum
: Request type.

COMPILE_BLOCK_END

Last instruction of basic block translated. Fields:

VADDR unsigned int
: Address of next instruction after this basic block

COMPILE_BLOCK_START

First instruction of basic block translated. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VADDR unsigned int
: Address of first instruction in basic block

COMPILE_INST

ARM instruction compiled. Fields:

DISASS string
: Disassembly of the instruction.

ISET enum
: The instruction set of this instruction.

ITSTATE unsigned int
: The ITSTATE current for the instruction.

OPCODE unsigned int
: The opcode of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

CONTEXTIDR

Context ID Register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

NS bool
: Secure or nonsecure banked register is accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

CONTEXT_SYNC

Called for every context synchronization event. Fields:

ARCH_CSE bool
: Architectural Context Synchronization Event

SYSREG_SYNC bool
: System register synchronization event

CORE_ENDIAN

Core BE8 Big-Endian state changed. Fields:

BE8 bool
: Core BE8 Big-Endian state

CORE_INFO

Static processor attributes. Only triggered by a call to DumpState(). Fields:

ARCH_PROFILE enum
: The architecture profile of the core.

CLUSTER_ID unsigned int
: The cluster ID of this processor.

CORE_NUM unsigned int
: The number of this core in an MP processor.

DETAILED_CORE_FEATURES enum
: Detailed core features as a bitfield. Enum values are bitmasks to extract the info from the field

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

FPU_VERSION enum
: The VFP version implemented by the core.

MEM_ARCH enum
: The memory architecture of the core.

MIDR_EL1 unsigned int
: MIDR_EL1 value

NUM_CORES unsigned int
: The number of cores in this MP processor.

QUANTUM_SIZE unsigned int
: The default quantum size of the core.

SECURITY_FEATURES bool
: Does the core have security features?

CORE_LOADS

Processor load accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_REGS

Changes of the core registers R0 to R14. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number, 0 to 14.

MODE enum
: Bank of the register accessed.

OLD_VALUE unsigned int
: The old value overwritten.

PHYS_ID enum
: The physical register accessed.

VALUE unsigned int
: The new value written to the register.

CORE_REGS64

Changes of the core registers X0..X30, SP_ELn. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: The register number, 0..30 for X0..X30, >=32 for SP_ELn.

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

CORE_STORES

Processor store accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CP14_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP14_WRITE

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

CP15_WRITE

System Control Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

CPSR

CPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

OLD_VALUE unsigned int
: The old CPSR value

UNKNOWN unsigned int
: Bits within the register that have unknown value.

VALUE unsigned int
: The new CPSR value

CRYPTO_SPEC

Every crypto instruction speculatively executed.

CorePowerStateChange

Core power-up state has changed. Fields:

PU bool
: Whether the core powered up

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DATA_CACHE_SET_ALLOC_TAGS

Set allocation tags from DC GVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of memory in bytes.

TAG unsigned int
: The value of the tag being set.

VADDR unsigned int
: Virtual address used for setting allocation tags.

DATA_CACHE_ZERO

Zero and invalidate cache line from DC ZVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of zero'd memory in bytes.

VADDR unsigned int
: Virtual address of the zero'd cache line.

DEBUG_EVENT

Hardware debug support event. Fields:

EVENT enum
: Description of event

VALUE unsigned int
: data value

END_COMPILE

Compilation end. Fields:

END_OF_PAGE_COUNT unsigned int
: Number of basic blocks exited due to page boundary since START_COMPILE.

FETCHFAIL_COUNT unsigned int
: Number of basic blocks exited due to ifetch failure START_COMPILE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: Number of instructions compiled since START_COMPILE.

NONSEQ_COUNT unsigned int
: Number of basic blocks exited due to non-sequential instructions since START_COMPILE.

PAGE_STRADDLE_COUNT unsigned int
: Number of basic blocks exited due to unaligned instructions crossing page since START_COMPILE.

ESB

Error Synchronization Barrier hit. Fields:

ASYNC bool
: ESB is async

IMPLICIT bool
: ESB is implicit

EXCEPTION

Exception taken. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

IS_PHYSICAL bool
: Physical or Virtual exception

LR unsigned int
: The value assigned to the link register.

PC unsigned int
: The location where the exception occurred.

PREFERRED_RETURN unsigned int
: The preferred return address for the exception.

TARGET_ISET enum
: The instruction set of the exception handler code.

TARGET_PC unsigned int
: The address the exception branches to.

VECTOR enum
: The exception vector.

VECTOR_OFFSET unsigned int
: The offset of an exception vector from the base address.

EXCEPTION_END

Every exception completed.

EXCEPTION_RAISE

Every exception raised.

EXCEPTION_RETURN

Branches on leaving exception. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

EXCEPTION_RETURN_PREBRANCH

Exception return event that must occur before PC and processor state have updated.

EXCEPTION_START

Every exception started.

FIQ_TAKEN

FIQ taken exception.

FP_STATE

Floating point state. Fields:

FPSR unsigned int
: FPSR register value.

MASK unsigned int
: The mask indicating updated cumulative bits.

HLT

HLT instruction occurred. Fields:

IMM16 unsigned int
: The 16-bit immediate value encoded in HLT instruction.

INST

Every instruction executed. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DEBUG_STATE bool
: The instruction is executed in debug state.

DISASS string
: Disassembly of instruction.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The current instruction set.

ITSTATE unsigned int
: The current ITSTATE.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

MODE enum
: The mode the core is in.

NS unsigned int
: The core's non-secure bit.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

OPCODE unsigned int
: The opcode of the instruction.

PADDR unsigned int
: The physical address of the instruction.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

INST_END

Every instruction completed.

INST_START

Every instruction started. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

INST_STRADDLE

Instruction straddles boundary. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

IRQ_TAKEN

IRQ taken exception.

ISB

ISB instruction occurred.

LOAD_MULTIPLE_REGS_LIST

List of destination registers (not source) of a multiple registers load instruction. Fields:

REG string
: Register to add into the list

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOCAL_MONITOR

Local monitor activity. Fields:

PADDR unsigned int
: Local Monitor Address

State enum
: State of the monitor (Open/Exclusive)

MEMMAP_DEBUG_READ

Memory mapped reads to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

MEMMAP_DEBUG_WRITE

Memory mapped writes to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

MMU_TRANS

Address translation information. Fields:

ASID unsigned int
: Address space identifier.

CORE_NUM unsigned int
: Core number in a multi processor.

Hyp bool
: Entry matches in Hyp state only

INNERCACHE_RA bool
: Is the inner cache allocate on read

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INNERCACHE_TRANSIENT bool
: Is the inner write-through transient

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

MEMTYPE enum
: Memory type.

NSDESC enum
: Is secure side supposed to access secure or nonsecure memory

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TRANSIENT bool
: Is the outer write-through transient

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

PADDR unsigned int
: Physical address of the access.

PAGESIZE unsigned int
: Page size as log2(size).

SH enum
: Shareability

SIDE enum
: Inst / Data.

STAGE1_PERM unsigned int
: Stage 1 permissions, [5:3] Privileged access (XWR) [2:0] User access (XWR)

STAGE2_PERM unsigned int
: Stage 2 permission mask [5:3] Privileged access (XWR) [2:0] User access (XWR)

VADDR unsigned int
: Virtual address of the access.

VMID unsigned int
: Virtual machine identifier.

XS bool
: XS Attribute

nG enum
: Flag indicating whether ASID will be matched

MMU_TTB_READ

This event is triggered by reads caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data read.

IPA unsigned int
: For stage 1, the IPA of the read.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OUTADDR unsigned int
: The output physical address.

PADDR unsigned int
: The physical address of the read.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MMU_TTB_WRITE

This event is triggered by writes caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data written.

IPA unsigned int
: For stage 1, the IPA of the write.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

PADDR unsigned int
: The physical address of the write.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MODE_CHANGE

Mode change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: The new mode.

NON_SECURE enum
: The new security state

OLD_MODE enum
: The old mode.

NOT_TAG_CHECKED_INST

MTE Tag not checked for current instruction.

PERIODIC

Called for every quantum. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PC unsigned int
The address of the next instruction to be executed on this CPU.

PMU_COUNTER_OVERFLOW

PMU counter overflow. Fields:

EVENT_ID unsigned int
Event Identifier
INDEX unsigned int
Counter index (as selected by PMSELR).
INTERRUPT bool
Is interrupt enabled on overflow for this counter

PREFETCH_MEMORY64

Prefetch from PRFM or PRFUM instructions. Fields:

PRFOP unsigned int
Prefetch hint
VADDR unsigned int
Virtual address of the location that should be prefetched.

PRELOAD_DATA

Data preload from PLD instruction. Fields:

VADDR unsigned int
Virtual address of the data that should be preloaded.

PRELOAD_INST

Instruction preload from PLI instruction. Fields:

VADDR unsigned int
Virtual address of the instruction that should be preloaded.

PRE_CORE_LOAD

Trace just before a core load. Fields:

CURRENT_TIME unsigned int
The core's current time, as simulated time plus local time.
LOCAL_TIME unsigned int
The core's local time, relative to the current quantum.

PRE_CORE_STORE

Trace just before a core store. Fields:

CURRENT_TIME unsigned int
The core's current time, as simulated time plus local time.
LOCAL_TIME unsigned int
The core's local time, relative to the current quantum.

PRE_TTB_READ

This event is triggered before reads caused by a translation table walk. Fields:

ADDR unsigned int
The physical address of the read.

PSTATE_PACTIVE_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions on PChannel. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

PSTATE_PACTIVE_OPMODE_TRANSITION_DENIED

Trace the denied pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_POWER_TRANSITION

Trace pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

PSTATE_PACTIVE_POWER_TRANSITION_DENIED

Trace the denied pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_THREAD_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions for threads on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

RUN_STATE

Run state transition. Fields:

INST_COUNT unsigned int
: Ticks count at point of transition.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NEW enum
: New run state.

OLD enum
: Old run state.

SEMIHOSTING_CALL

Call of a semihost function occurred. Fields:

PC unsigned int
: The program counter after the semihosting call.

SEMIHOSTING_PRECALL

About to call semihost. Fields:

EL2 bool
: Use EL2 translation regime.

NS bool
: Is Non-Secure.

REG_WIDTH unsigned int
: The current register width in bytes.

SIGNAL

External signal state change. Fields:

SIGNAL enum
: Signal that changed

STATE bool
: Signal asserted state

SOFTWARE_STEP

Return the Debug Software Step state. Fields:

STATE enum
: Software Step state

SPSR

SPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: Which of the banked SPSR registers is written.

OLD_VALUE unsigned int
: The old SPSR value

VALUE unsigned int
: The new SPSR value

START_COMPILE

Compilation started. Fields:

VADDR unsigned int
: Instruction where compilation begins.

STG_MULTIPLE_MICRO_OPS_ADDRESS_RANGE

Address range for MTE stores with multiple uops. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

STORE_MULTIPLE_REGS_ADDRESS_RANGE

Address range for stores with multiple registers. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

SUBOPTIMAL_LDST_RETIRED

suboptimal load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1: half-word, 2: word ...)

OPTIMAL_ALIGN_SELECTOR unsigned int
: Selector to decide the optimal alignment size (1: access width, 2: 4-bytes, 3: 8-bytes, ..., 12: 4-kbytes)

VADDR unsigned int
: The virtual address of the access

SYNC

Called for every synchronization, that is every quantum break. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

LOCAL_QUANTUM unsigned int
: The local quantum of this CPU.

LOCAL_TIME unsigned int
: The local time of this CPU.

SYNC_REASON enum
: Reason for sync.

SYSCALL

System call instruction executed. Fields:

IMM unsigned int
: Immediate value of the system call instruction.

TYPE enum
: System call type.

VADDR unsigned int
: Instruction that caused the system call.

SYSREG_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE32

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE64

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

UNALIGNED_LDST_RETIRED

Processor unaligned load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1=>half-word, 2->word ...)

NUMBER_OF_BEATS unsigned int
: Number of accesses (beats) in this burst (Total size in bytes is ACCESS_SIZE * NUMBER_OF_BEATS)

VADDR unsigned int
: The virtual address of the access

VFP_D_REGS

VFP/NEON D 64 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_Q_REGS

VFP/NEON Q 128 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding byte in the register. For example, 0x1 would indicate a write to VALUE[7:0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_SYS_REGS

Writes to the VFP/NEON units system registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: Which VFP system register is written.

MASK unsigned int
: The mask indicating updated cumulative bits.

OLD_VALUE unsigned int
: The register's old value overwritten.

VALUE unsigned int
: The new value written to the VFP system register.

VFP_S_REGS

VFP/NEON S 32 bit register write. Fields:

ALIASING enum
: Type of register aliasing used.

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

WAYPOINT

Signals end of basic block execution. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: Origin instructions set

IS_COND bool
: Indicates if this is a conditional waypoint.

PC unsigned int
: Origin address (or 0 if unavailable)

TAKEN bool
: Indicates if this waypoint was taken.

TARGET unsigned int
: Destination address

TARGET_ISET enum
: Destination instructions set

WFE_END

WFE ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFE.

WFE_EVENT_REGISTER

WFE event register status: set/clear, reason. Fields:

INST_COUNT unsigned int
: Ticks count.

REASON enum
: Reason for set/clear. Only REASON==0 (Cleared by WFE) clears the bit, all other reasons set it.

WFE_IGNORED

WFE ignored. Fields:

EVENT bool
: This WFE was ignored because the event register was set.

INST_COUNT unsigned int
: Ticks count when ignoring WFE.

TRAPPED bool
: This WFE was trapped.

WFE_START

WFE entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFE.

WFI_END

WFI ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFI.

WFI_IGNORED

WFI ignored. Fields:

DISABLED bool
: This WFI was ignored because WFI is disabled.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: Ticks count when ignoring WFI.

REASON enum
: specifies reason why WFI trace was ignored

TRAPPED bool
: This WFI was trapped.

WFI_START

WFI entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFI.

WFI_WAKEUP

WFI wakeup. Fields:

INST_COUNT unsigned int
: Ticks count when WFI wakeup occurred.

REASON enum
: Reason for wakeup.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.4 ARM_AEMv8-A_MP - trace

This section describes the trace sources.

ASYNC_MEMORY_FAULT

Context ID Register write. Fields:

FAULT unsigned int
: Fault status in ESR format

PADDR unsigned int
: Physical Address (or 0 if unavailable)

VADDR signed int
: Virtual Address (or 0 if unavailable)

ATOMIC_END_ACCESS

Bus trace access for atomics. Fields:

ACCESS_FAIL bool
: Memory access failed

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

LOAD_VALUE unsigned int
: Loaded values

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ATOMIC_START_ACCESS

Bus trace access for atomics. Fields:

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

NSDESC unsigned int
: The security state of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ArchMsg.Error.exit_code

Fields:

EXITCODE signed int
: Exit code to be returned

KIND string
: Component kind that invoked the exit code trace

REASON string
: Reason why the exit code trace was invoked (optional)

ArchMsg.Warning.IgnoredWarmResetRequest

Event fires when software attempts to request warm reset through write to deprecated DBGPRCR.CWRR.

ArchMsg.Warning.TarmacReconstructor

Unexpected event happened during Tarmac reconstruction. Fields:

REASON string
: Reason

ArchMsg.Warning.Unpredictably Indexed PM Event Register

Fields:

IsDirect unsigned int
: Direct Access rather than indirect

N unsigned int
: Number of Accessible Registers

SEL unsigned int
: Selector

ArchMsg.Warning.branch_to_unaligned_address

Fields:

ADDR unsigned int
: Unaligned branch address

ArchMsg.Warning.cache_contents_unknown

A cache has been activated without being invalidated and may contain UNKNOWN content. Fields:

REGIME unsigned int
: Current translation regime

SIDE unsigned int
: 0 d-side cache or unified cache, 1 i-side

ArchMsg.Warning.dcimvac_matches_watchpoint

Watchpoints matching an AArch32 DCIMVA are implementation defined.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_fieldmismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_fracbitsnegative

VCVT Instruction is unpredictable with a negative number of fraction bits.

ArchMsg.Warning.decode_initblock

Instruction is unpredictable in an IT block.

ArchMsg.Warning.decode_initblocknotlast

Instruction is unpredictable in an IT block when not the last.

ArchMsg.Warning.decode_invalid_condition

Instruction is unpredictable when the condition is not AL.

ArchMsg.Warning.decode_invalidfieldcombination

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_invalidvalue

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_mem_hint_unallocated

There is no memory hint allocated to this bit pattern.

ArchMsg.Warning.decode_registermatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registermismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registeroutofrange

Float point instruction is unpredictable indexing beyond the end of the register bank.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_sbzsbo

Reserved bits in the instruction are not canonical.

ArchMsg.Warning.decode_transactiontoobig

Attempt to perform a memory transaction of more than 128 bytes.

ArchMsg.Warning.decode_unpred_other

Instruction is unpredictable.

ArchMsg.Warning.decode_unpreduseofpc

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13inreglist

Instruction is unpredictable with R13 in register-list.

ArchMsg.Warning.decode_useofd16orhigher

Attempt to use register D16 or higher where it is not allowed.

ArchMsg.Warning.decode_writebackandbaseinlist

Instruction is unpredictable with write back when the base register is in the transfer list.

ArchMsg.Warning.decode_zeroregistersinlist

Instruction is unpredictable when no registers are to be transfered.

ArchMsg.Warning.recursive_branch

An instruction is performing a branch that targets the same instruction. If this was intended then a WFI might be more effective. Fields:

PC unsigned int
: Address of instruction causing the branch

ArchMsg.Warning.recursive_exception

An instruction has generated an exception that targets the same instruction. Fields:

CPSR unsigned int
: Processor state

PC unsigned int
: Address of instruction causing the exception

TYPE enum
: Type of exception

ArchMsg.Warning.reentrant_vector_catch

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR unsigned int
vector address

ArchMsg.Warning.reserved_it_state

Fields:

ITSTATE unsigned int
reserved IT state

ArchMsg.Warning.tlb_contents_unknown

Translation has been enabled without invalidating the TLB. Fields:

EL unsigned int
Exception level of translation regime
INVALIDITY unsigned int
Which TLB may be invalid (if TLBs are separate)
NON_SECURE unsigned int
Security state of translation regime
VMID unsigned int
Current VMID, if applicable

ArchMsg.Warning.unknown_DLR_DSPSR

Exiting debug state with unknown DLR or DSPSR.

ArchMsg.Warning.unknown_ELR_SPSR

Returning from debug exception with unknown ELR or SPSR.

ArchMsg.Warning.unpredictable_a32_breakpoint

Fields:

ADDR1 unsigned int
Address
ADDR2 unsigned int
Breakpoint address
BAS unsigned int
BAS Field
IS_ADDRESS_MISMATCH_BREAKPOINT bool
Is Mismatch breakpoint, not a match breakpoint
N unsigned int
Breakpoint number
OFFSET unsigned int
Indicates which halfword actually considering into the A32 instruction

ArchMsg.Warning.unpredictable_banked_register_access

Fields:

R unsigned int
R bit
SYSM unsigned int
SYSm

ArchMsg.Warning.unpredictable_ccfail_undef

Fields:

OPCODE unsigned int
opcode of the instruction

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VADDR unsigned int
Virtual address of the instruction being executed

ArchMsg.Warning.unpredictable_ici_bits

ICI bits do not apply or are architecturally unpredictable for instruction.

ArchMsg.Warning.unpredictable_t32_breakpoint

Fields:

ADDR1 unsigned int
Address
ADDR2 unsigned int
Breakpoint address
BAS unsigned int
BAS Field
IS_ADDRESS_MISMATCH_BREAKPOINT bool
Is Mismatch breakpoint, not a match breakpoint
N unsigned int
Breakpoint number
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_t32_vector_catch

Fields:

ADDR unsigned int
Address
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_watchpoint_far

Fields:

LOWER_BOUND unsigned int
the lowest address accessed by the instruction that triggered the watchpoint
UPPER_BOUND unsigned int
the highest watchpointed address accessed by that instruction

ArchMsg.Warning.warning_AdvSIMDExpandImmUnexpectedZero

AdvSIMDExpandImm may treat this immediate value as UNPREDICTABLE.

ArchMsg.Warning.warning_ConditionalSMC

SMC instruction has UNPREDICTABLE effects when conditional when combined with traps.

ArchMsg.Warning.warning_access_crosses_page_boundary

Access crosses page boundary.

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault_on_both_pages

Access crosses page boundary has fault(s).

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_access_crosses_page_boundary_spanning_different_memory

Fields:

ADDR unsigned int
: address of access crossing page

MEMTYPE_PAGE1 signed int
: First Page's memory type (0-Normal, 1-Non-Normal)

MEMTYPE_PAGE2 signed int
: Second Page's memory type (0-Normal, 1-Non-Normal)

ArchMsg.Warning.warning_access_wraps_around_memory

Fields:

WIDTH unsigned int
: Address width

ArchMsg.Warning.warning_bx_from_thumbee

Fields:

ADDR unsigned int
: Destination address

ArchMsg.Warning.warning_ccsidr_unimplemented_level

Fields:

CSSELR unsigned int
: current effective value of CSSELR

ArchMsg.Warning.warning_change_to_ns_when_tge_set

Attempting to change NS to 1 when HCR.TGE = 1.

ArchMsg.Warning.warning_contiguous_bit_check_abort

Fields:

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ArchMsg.Warning.warning_contiguous_bit_error

Fields:

CONTIG_ADDR unsigned int
: address of the TLB first read, with which this entry is expected to be contiguous

CONTIG_DATA unsigned int
: expected contents based on the entry at CONTIG_ADDR

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ENTRY_DATA unsigned int
: contents of conflicting TLB entry

ArchMsg.Warning.warning_cp10_cp11_mismatch

Fields:

EL unsigned int
: EL owning the control

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_cp10_cp11_reserved_value

Fields:

CP10 unsigned int
: CP10 access

CP11 unsigned int
: CP11 access

ArchMsg.Warning.warning_csselr_level_out_of_range

Fields:

IMPLEMENTED unsigned int
: number of cache levels implemented

LEVEL unsigned int
: written value of CSSELR.Level

TYPE enum
: The type of cache selected by CSSELR.InD and TnD

ArchMsg.Warning.warning_debug_flow_control_bits_not_obeyed

Fields:

ERROR enum
: type of error resulting

REGISTER enum
: register under consideration

ArchMsg.Warning.warning_debug_register_access_during_reset

Fields:

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

ArchMsg.Warning.warning_decode_cps_inconsistent_fields

The mode setting fields in the CPS instruction are inconsistent.

ArchMsg.Warning.warning_decode_invalid_state

Instruction is unpredictable in current exception-level/security state.

ArchMsg.Warning.warning_default_cacheable_mmu_on

Default cacheable is enabled (HCR.DC = 1) while MMU is enabled.

ArchMsg.Warning.warning_default_cacheable_vmmu_off

Default cacheable is enabled (HCR.DC = 1) while HCR.VM = 0.

ArchMsg.Warning.warning_deprecated_wvr_bit_2_set

Fields:

N unsigned int
: index

WVR unsigned int
: DBGWVR

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_eret_while_software_step_active_pending

Missing ISB between setting MDSCR_EL1.SS and ERET.

ArchMsg.Warning.warning_exclusive_to_non_normal

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_exclusive_to_non_writeback

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_execute_from_device_memory

An attempt was made to execute from Device memory.

ArchMsg.Warning.warning_illegal_cpsr_mode

Fields:

NEW_MODE unsigned int
: New value of CPSR.M

ArchMsg.Warning.warning_illegal_srs_mode

Fields:

MODE unsigned int
: mode for Banked SP

ArchMsg.Warning.warning_implementation_defined_read_debug_register_in_SCS

Read Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_implementation_defined_sequential_security_transitions_supported

The behaviour of a sequential instruction fetches that cross from non-secure to secure memory and contain SG instruction is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_implementation_defined_stack_limit_check_supported

It is IMPLEMENTATION DEFINED whether stack pointer limit checking is performed for the this instructions. Fields:

ADDRESS unsigned int
: Address of instruction

OPCODE unsigned int
: instruction opcode

ArchMsg.Warning.warning_implementation_defined_write_debug_register_in_SCS

Write Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

DATA unsigned int
: data attempted to be written

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_invalid_tcr_granule

Fields:

REQUEST signed int
: page size requested (or 0 for reserved)

SUBSTITUTE signed int
: best guess available page size

TG_ID bool
: bits TG0 or TG1

ArchMsg.Warning.warning_load_multiple_user_registers_from_user_mode

An LDM(user registers) instruction executed from user mode.

ArchMsg.Warning.warning_load_pc_from_unaligned

PC loaded from an unaligned location.

ArchMsg.Warning.warning_mte_in_mem_attr_not_supported

Fields:

ATTR unsigned int
: Memory attribute

ArchMsg.Warning.warning_reserved_breakpoint_state_match

Fields:

IS_BREAKPOINT unsigned int
: Is breakpoint, not watchpoint

ArchMsg.Warning.warning_shareability

Unpredictable: combination of the 1st and 2nd stages of translation is Normal Inner Non-Cacheable, Outer Non-Cacheable.

ArchMsg.Warning.warning_software_step_set_while_enabled

MDSCR_EL1.SS set to 1 while software step debug exceptions are enabled.

ArchMsg.Warning.warning_thumb_instruction_wraps_around_memory

Thumb instruction warps around memory.

ArchMsg.Warning.warning_ttbr_sbz_bits_are_not_zero

Fields:

MSB unsigned int
: x-1

TTBR enum
: which TTBR register

ArchMsg.Warning.warning_unaligned_address_dbgdtrrx_write

Fields:

ADDR unsigned int
: Address

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unaligned_address_dbgdtrtx_read

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_unaligned_to_device

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_unaligned_to_strongly_ordered

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_unknown_sau_rnr

SAU_RNR was set to an unsupported value.

ArchMsg.Warning.warning_unpredictable_AIRCR_PRIP_and_BFHFNMINP

The effect of setting both AIRCR.BFHFNMINP and AIRCR.PRIP to 1 is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTCLRACTIVE_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTCLRACTIVE if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_and_SYSRESETREQ

When the processor is halted in Debug state, if a write to the register writes a 1 to both VECTRESET and SYSRESETREQ, the behavior is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTRESET if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_incorrect_VKEY

The value 0x05FA must be written to AIRCR.VKEY, otherwise the register write is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_EXE_RETURN_Reserved_Bit

EXC_RETURN[23:7] are reserved with the special condition that all bits should be written as one. Values other than all 1s are UNPREDICTABLE. Fields:

EXC_RETURN unsigned int
: Special PC value

SBO_MASK unsigned int
: Which bits Should Be One to not UNPRED

ArchMsg.Warning.warning_unpredictable_change_to_DEMCR_MON_STEP_at_insufficient_priority

The effect of changing DEMCR.MON_STEP at an execution priority that is lower than the priority of the DebugMonitor exception is CONSTRAINED UNPREDICTABLE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_change_to_priority_of_active_exception

Changing the priority of an active exception is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_clear_lspact

Fields:

DRVEXC_TAKEN unsigned int
: derived exception taken not pended

REACHED_LAST_STORE unsigned int
: only the last faulted

ArchMsg.Warning.warning_unpredictable_dbg_exit_unaligned_dlr

Debug exit to AArch32 with DLR[0] set to 1. PC[0] can be set to 0 or to DLR[0].

ArchMsg.Warning.warning_unpredictable_exception_catch

Generation of an exception catch event is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_inconsistent_state

state on exception return is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_instruction

exception return instruction is unpredictable.

ArchMsg.Warning.warning_unpredictable_in_debug_state

Instruction is UNPREDICTABLE when executed in debug state.

ArchMsg.Warning.warning_unpredictable_mair_encoding

Fields:

ATTR unsigned int
: Memory attribute

ArchMsg.Warning.warning_unpredictable_pmu_counter_access

Access to PMU counters from non-secure EL0 or EL1 is UNPREDICTABLE with MDCR_EL2.HPMN set to 0.

ArchMsg.Warning.warning_unpredictable_prioritization_breakpoint_match_vector_catch

Two events with the same priority occurred at the same instruction: Address Matching Vector Catch debug event, and Breakpoint debug event It is constrained unpredictable which is taken.

ArchMsg.Warning.warning_unpredictable_stack_selection

UNPREDICTABLE state on context switch.

ArchMsg.Warning.warning_unpredictable_tsize_out_of_range

Fields:

TCR enum
: Which TCR

TSIZE signed int
: Value of TSize

TSIZE_MAX signed int
: The maximum allowed value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

TTBR unsigned int
Which TTBR region

ArchMsg.Warning.warning_unpredictable_unaligned_pc_as_base_register

The PC value must be word-aligned, otherwise the behavior of the instruction is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_unaligned_pop_stack

Fields:

FRAME_ADDR unsigned int
unaligned stack frame address

ArchMsg.Warning.warning_unpredictable_vmsa_memattrib

Fields:

TEXCB unsigned int
texcb value
TEX_REMAP bool
use tex remap

ArchMsg.Warning.warning_unpredictable_vtcr_t0sz_sl0

Fields:

SL0 signed int
VTCR.SL0, starting level for stage 2 translation table walks
T0SZ signed int
VTCR.T0SZ, required input address range

ArchMsg.Warning.warning_unpredictable_write_DHCSR

Fields:

NEW unsigned int
data attempted to be written
OLD unsigned int
previous value

ArchMsg.Warning.warning_unpredictable_write_SHCSR

UNPREDICTABLE setting a pending bit to 1 for an exception with priority >= execution priority.

ArchMsg.Warning.warning_unsupported_access_to_memory_mapped_register

Access to memory mapped register is unsupported.

ArchMsg.Warning.warning_user_jmcr_access

Fields:

WRITE unsigned int
access is write

ArchMsg.Warning.warning_wcr_mask_and_bas

DBGWCR_EL1.MASK is non-zero and BAS is not 0xff.

ArchMsg.Warning.warning_wcr_mask_reserved

DBGWCR_ELn.MASK is set to a reserved value.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_wcr_non_configuous_bas

DBGWCR_EL1.BAS has non-contiguous set of ones.

ArchMsg.Warning.warning_wvr_masked_bit_not_zero

DBGWVRn_EL1.VA contains a bit masked by DBGWCRn_EL1.MASK that is not zero.

ArchMsg.Why.why_branch_target_exception

A branch target exception has occurred. Fields:

BTYPE unsigned int
: pstate.BTYPE at time branch taken

OPCODE unsigned int
: opcode at branch target

PC unsigned int
: Virtual Address of branch target

ArchMsg.Why.why_illegal_state

An illegal mode change has occured. Fields:

MESSAGE string
: The message

BRANCH_MISPREDICT

Simulating branch mispredict. Fields:

PC unsigned int
: Origin address (or 0 if unavailable)

BRA_DIR

Direct branches, to immediate address. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR

Indirect branches, perhaps to a register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR_CCFAIL

Indirect branches fail, perhaps to a register.

BRA_RET_CCFAIL

Branche return fail, perhaps to a register.

CACHE_MAINTENANCE_OP

Cache Maintenance Operation. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Specified MVA or Set/way

FUNCTION enum
: Clean / Invalidate

SCOPE enum
: Affected region

SIDE enum
: Inst / Data.

CCFAIL

Conditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCFAIL_UNC

Unconditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS

Conditional instruction condition check pass. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS_UNC

Unconditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CHECKPOINT_MESSAGE

Report error messages from the checkpointing process. Fields:

message string
: Message contents.

CHECKPOINT_RESTORE_END

Checkpoint restore completed.

CHECKPOINT_RESTORE_START

Checkpoint restore about to start.

CHECKPOINT_SAVE_END

Checkpoint save completed.

CHECKPOINT_SAVE_START

Checkpoint save about to start.

CODE_CACHE_MAINT

Code cache maintenance. Fields:

END_INCL unsigned int
: Inclusive end addres for by VA or PA requests

NS bool
: Non-secure world for PA

START unsigned int
: Start address for by VA or PA requests

TYPE enum
: Request type.

COMPILE_BLOCK_END

Last instruction of basic block translated. Fields:

VADDR unsigned int
: Address of next instruction after this basic block

COMPILE_BLOCK_START

First instruction of basic block translated. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VADDR unsigned int
: Address of first instruction in basic block

COMPILE_INST

ARM instruction compiled. Fields:

DISASS string
: Disassembly of the instruction.

ISET enum
: The instruction set of this instruction.

ITSTATE unsigned int
: The ITSTATE current for the instruction.

OPCODE unsigned int
: The opcode of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

CONTEXTIDR

Context ID Register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

NS bool
: Secure or nonsecure banked register is accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

CONTEXT_SYNC

Called for every context synchronization event. Fields:

ARCH_CSE bool
: Architectural Context Synchronization Event

SYSREG_SYNC bool
: System register synchronization event

CORE_ENDIAN

Core BE8 Big-Endian state changed. Fields:

BE8 bool
: Core BE8 Big-Endian state

CORE_INFO

Static processor attributes. Only triggered by a call to DumpState(). Fields:

ARCH_PROFILE enum
: The architecture profile of the core.

CLUSTER_ID unsigned int
: The cluster ID of this processor.

CORE_NUM unsigned int
: The number of this core in an MP processor.

DETAILED_CORE_FEATURES enum
: Detailed core features as a bitfield. Enum values are bitmasks to extract the info from the field

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

FPU_VERSION enum
: The VFP version implemented by the core.

MEM_ARCH enum
: The memory architecture of the core.

MIDR_EL1 unsigned int
: MIDR_EL1 value

NUM_CORES unsigned int
: The number of cores in this MP processor.

QUANTUM_SIZE unsigned int
: The default quantum size of the core.

SECURITY_FEATURES bool
: Does the core have security features?

CORE_LOADS

Processor load accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_REGS

Changes of the core registers R0 to R14. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number, 0 to 14.

MODE enum
: Bank of the register accessed.

OLD_VALUE unsigned int
: The old value overwritten.

PHYS_ID enum
: The physical register accessed.

VALUE unsigned int
: The new value written to the register.

CORE_REGS64

Changes of the core registers X0..X30, SP_ELn. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: The register number, 0..30 for X0..X30, >=32 for SP_ELn.

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

CORE_STORES

Processor store accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CP14_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP14_WRITE

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

CP15_WRITE

System Control Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

CPSR

CPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

OLD_VALUE unsigned int
: The old CPSR value

UNKNOWN unsigned int
: Bits within the register that have unknown value.

VALUE unsigned int
: The new CPSR value

CRYPTO_SPEC

Every crypto instruction speculatively executed.

CorePowerStateChange

Core power-up state has changed. Fields:

PU bool
: Whether the core powered up

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DATA_CACHE_SET_ALLOC_TAGS

Set allocation tags from DC GVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of memory in bytes.

TAG unsigned int
: The value of the tag being set.

VADDR unsigned int
: Virtual address used for setting allocation tags.

DATA_CACHE_ZERO

Zero and invalidate cache line from DC ZVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of zero'd memory in bytes.

VADDR unsigned int
: Virtual address of the zero'd cache line.

DEBUG_EVENT

Hardware debug support event. Fields:

EVENT enum
: Description of event

VALUE unsigned int
: data value

END_COMPILE

Compilation end. Fields:

END_OF_PAGE_COUNT unsigned int
: Number of basic blocks exited due to page boundary since START_COMPILE.

FETCHFAIL_COUNT unsigned int
: Number of basic blocks exited due to ifetch failure START_COMPILE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: Number of instructions compiled since START_COMPILE.

NONSEQ_COUNT unsigned int
: Number of basic blocks exited due to non-sequential instructions since START_COMPILE.

PAGE_STRADDLE_COUNT unsigned int
: Number of basic blocks exited due to unaligned instructions crossing page since START_COMPILE.

ESB

Error Synchronization Barrier hit. Fields:

ASYNC bool
: ESB is async

IMPLICIT bool
: ESB is implicit

EXCEPTION

Exception taken. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

IS_PHYSICAL bool
: Physical or Virtual exception

LR unsigned int
: The value assigned to the link register.

PC unsigned int
: The location where the exception occurred.

PREFERRED_RETURN unsigned int
: The preferred return address for the exception.

TARGET_ISET enum
: The instruction set of the exception handler code.

TARGET_PC unsigned int
: The address the exception branches to.

VECTOR enum
: The exception vector.

VECTOR_OFFSET unsigned int
: The offset of an exception vector from the base address.

EXCEPTION_END

Every exception completed.

EXCEPTION_RAISE

Every exception raised.

EXCEPTION_RETURN

Branches on leaving exception. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

EXCEPTION_RETURN_PREBRANCH

Exception return event that must occur before PC and processor state have updated.

EXCEPTION_START

Every exception started.

FIQ_TAKEN

FIQ taken exception.

FP_STATE

Floating point state. Fields:

FPSR unsigned int
: FPSR register value.

MASK unsigned int
: The mask indicating updated cumulative bits.

HLT

HLT instruction occurred. Fields:

IMM16 unsigned int
: The 16-bit immediate value encoded in HLT instruction.

INST

Every instruction executed. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DEBUG_STATE bool
: The instruction is executed in debug state.

DISASS string
: Disassembly of instruction.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The current instruction set.

ITSTATE unsigned int
: The current ITSTATE.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

MODE enum
: The mode the core is in.

NS unsigned int
: The core's non-secure bit.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

OPCODE unsigned int
: The opcode of the instruction.

PADDR unsigned int
: The physical address of the instruction.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

INST_END

Every instruction completed.

INST_START

Every instruction started. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

INST_STRADDLE

Instruction straddles boundary. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

IRQ_TAKEN

IRQ taken exception.

ISB

ISB instruction occurred.

LOAD_MULTIPLE_REGS_LIST

List of destination registers (not source) of a multiple registers load instruction. Fields:

REG string
: Register to add into the list

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOCAL_MONITOR

Local monitor activity. Fields:

PADDR unsigned int
: Local Monitor Address

State enum
: State of the monitor (Open/Exclusive)

MEMMAP_DEBUG_READ

Memory mapped reads to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

MEMMAP_DEBUG_WRITE

Memory mapped writes to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

MMU_TRANS

Address translation information. Fields:

ASID unsigned int
: Address space identifier.

CORE_NUM unsigned int
: Core number in a multi processor.

Hyp bool
: Entry matches in Hyp state only

INNERCACHE_RA bool
: Is the inner cache allocate on read

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INNERCACHE_TRANSIENT bool
: Is the inner write-through transient

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

MEMTYPE enum
: Memory type.

NSDESC enum
: Is secure side supposed to access secure or nonsecure memory

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TRANSIENT bool
: Is the outer write-through transient

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

PADDR unsigned int
: Physical address of the access.

PAGESIZE unsigned int
: Page size as log2(size).

SH enum
: Shareability

SIDE enum
: Inst / Data.

STAGE1_PERM unsigned int
: Stage 1 permissions, [5:3] Privileged access (XWR) [2:0] User access (XWR)

STAGE2_PERM unsigned int
: Stage 2 permission mask [5:3] Privileged access (XWR) [2:0] User access (XWR)

VADDR unsigned int
: Virtual address of the access.

VMID unsigned int
: Virtual machine identifier.

XS bool
: XS Attribute

nG enum
: Flag indicating whether ASID will be matched

MMU_TTB_READ

This event is triggered by reads caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data read.

IPA unsigned int
: For stage 1, the IPA of the read.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OUTADDR unsigned int
: The output physical address.

PADDR unsigned int
: The physical address of the read.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MMU_TTB_WRITE

This event is triggered by writes caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data written.

IPA unsigned int
: For stage 1, the IPA of the write.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

PADDR unsigned int
: The physical address of the write.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MODE_CHANGE

Mode change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: The new mode.

NON_SECURE enum
: The new security state

OLD_MODE enum
: The old mode.

NOT_TAG_CHECKED_INST

MTE Tag not checked for current instruction.

PERIODIC

Called for every quantum. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PC unsigned int
The address of the next instruction to be executed on this CPU.

PMU_COUNTER_OVERFLOW

PMU counter overflow. Fields:

EVENT_ID unsigned int
Event Identifier
INDEX unsigned int
Counter index (as selected by PMSELR).
INTERRUPT bool
Is interrupt enabled on overflow for this counter

PREFETCH_MEMORY64

Prefetch from PRFM or PRFUM instructions. Fields:

PRFOP unsigned int
Prefetch hint
VADDR unsigned int
Virtual address of the location that should be prefetched.

PRELOAD_DATA

Data preload from PLD instruction. Fields:

VADDR unsigned int
Virtual address of the data that should be preloaded.

PRELOAD_INST

Instruction preload from PLI instruction. Fields:

VADDR unsigned int
Virtual address of the instruction that should be preloaded.

PRE_CORE_LOAD

Trace just before a core load. Fields:

CURRENT_TIME unsigned int
The core's current time, as simulated time plus local time.
LOCAL_TIME unsigned int
The core's local time, relative to the current quantum.

PRE_CORE_STORE

Trace just before a core store. Fields:

CURRENT_TIME unsigned int
The core's current time, as simulated time plus local time.
LOCAL_TIME unsigned int
The core's local time, relative to the current quantum.

PRE_TTB_READ

This event is triggered before reads caused by a translation table walk. Fields:

ADDR unsigned int
The physical address of the read.

PSTATE_PACTIVE_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions on PChannel. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DEVPACTIVE `unsigned int`
: Integer representation of DEVPACTIVE[] signals

NEW_STATE `enum`
: Operating mode after the transition

OLD_STATE `enum`
: Operating mode before the transition

PSTATE_PACTIVE_OPMODE_TRANSITION_DENIED

Trace the denied pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE `unsigned int`
: Integer representation of DEVPACTIVE[] signals

NEW_STATE `enum`
: Operating mode after the transition

OLD_STATE `enum`
: Operating mode before the transition

REASON `string`
: Reason for the transition being denied

PSTATE_PACTIVE_POWER_TRANSITION

Trace pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE `unsigned int`
: Integer representation of DEVPACTIVE[] signals

NEW_STATE `enum`
: Power mode after the transition

OLD_STATE `enum`
: Power mode before the transition

PSTATE_PACTIVE_POWER_TRANSITION_DENIED

Trace the denied pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE `unsigned int`
: Integer representation of DEVPACTIVE[] signals

NEW_STATE `enum`
: Power mode after the transition

OLD_STATE `enum`
: Power mode before the transition

REASON `string`
: Reason for the transition being denied

PSTATE_PACTIVE_THREAD_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions for threads on PChannel. Fields:

DEVPACTIVE `unsigned int`
: Integer representation of DEVPACTIVE[] signals

NEW_STATE `enum`
: Operating mode after the transition

OLD_STATE `enum`
: Operating mode before the transition

RUN_STATE

Run state transition. Fields:

INST_COUNT `unsigned int`
: Ticks count at point of transition.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NEW enum
: New run state.

OLD enum
: Old run state.

SEMIHOSTING_CALL

Call of a semihost function occurred. Fields:

PC unsigned int
: The program counter after the semihosting call.

SEMIHOSTING_PRECALL

About to call semihost. Fields:

EL2 bool
: Use EL2 translation regime.

NS bool
: Is Non-Secure.

REG_WIDTH unsigned int
: The current register width in bytes.

SIGNAL

External signal state change. Fields:

SIGNAL enum
: Signal that changed

STATE bool
: Signal asserted state

SOFTWARE_STEP

Return the Debug Software Step state. Fields:

STATE enum
: Software Step state

SPSR

SPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: Which of the banked SPSR registers is written.

OLD_VALUE unsigned int
: The old SPSR value

VALUE unsigned int
: The new SPSR value

START_COMPILE

Compilation started. Fields:

VADDR unsigned int
: Instruction where compilation begins.

STG_MULTIPLE_MICRO_OPS_ADDRESS_RANGE

Address range for MTE stores with multiple uops. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

STORE_MULTIPLE_REGS_ADDRESS_RANGE

Address range for stores with multiple registers. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

SUBOPTIMAL_LDST_RETIRED

suboptimal load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1: half-word, 2: word ...)

OPTIMAL_ALIGN_SELECTOR unsigned int
: Selector to decide the optimal alignment size (1: access width, 2: 4-bytes, 3: 8-bytes, ..., 12: 4-kbytes)

VADDR unsigned int
: The virtual address of the access

SYNC

Called for every synchronization, that is every quantum break. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

LOCAL_QUANTUM unsigned int
: The local quantum of this CPU.

LOCAL_TIME unsigned int
: The local time of this CPU.

SYNC_REASON enum
: Reason for sync.

SYSCALL

System call instruction executed. Fields:

IMM unsigned int
: Immediate value of the system call instruction.

TYPE enum
: System call type.

VADDR unsigned int
: Instruction that caused the system call.

SYSREG_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE32

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE64

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

UNALIGNED_LDST_RETIRED

Processor unaligned load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1=>half-word, 2->word ...)

NUMBER_OF_BEATS unsigned int
: Number of accesses (beats) in this burst (Total size in bytes is ACCESS_SIZE * NUMBER_OF_BEATS)

VADDR unsigned int
: The virtual address of the access

VFP_D_REGS

VFP/NEON D 64 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_Q_REGS

VFP/NEON Q 128 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding byte in the register. For example, 0x1 would indicate a write to VALUE[7:0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_SYS_REGS

Writes to the VFP/NEON units system registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: Which VFP system register is written.

MASK unsigned int
: The mask indicating updated cumulative bits.

OLD_VALUE unsigned int
: The register's old value overwritten.

VALUE unsigned int
: The new value written to the VFP system register.

VFP_S_REGS

VFP/NEON S 32 bit register write. Fields:

ALIASING enum
: Type of register aliasing used.

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

WAYPOINT

Signals end of basic block execution. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: Origin instructions set

IS_COND bool
: Indicates if this is a conditional waypoint.

PC unsigned int
: Origin address (or 0 if unavailable)

TAKEN bool
: Indicates if this waypoint was taken.

TARGET unsigned int
: Destination address

TARGET_ISET enum
: Destination instructions set

WFE_END

WFE ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFE.

WFE_EVENT_REGISTER

WFE event register status: set/clear, reason. Fields:

INST_COUNT unsigned int
: Ticks count.

REASON enum
: Reason for set/clear. Only REASON==0 (Cleared by WFE) clears the bit, all other reasons set it.

WFE_IGNORED

WFE ignored. Fields:

EVENT bool
: This WFE was ignored because the event register was set.

INST_COUNT unsigned int
: Ticks count when ignoring WFE.

TRAPPED bool
: This WFE was trapped.

WFE_START

WFE entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFE.

WFI_END

WFI ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFI.

WFI_IGNORED

WFI ignored. Fields:

DISABLED bool
: This WFI was ignored because WFI is disabled.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: Ticks count when ignoring WFI.

REASON enum
: specifies reason why WFI trace was ignored

TRAPPED bool
: This WFI was trapped.

WFI_START

WFI entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFI.

WFI_WAKEUP

WFI wakeup. Fields:

INST_COUNT unsigned int
: Ticks count when WFI wakeup occurred.

REASON enum
: Reason for wakeup.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.5 ARM_AEMv8M - trace

This section describes the trace sources.

ASYNC_MEMORY_FAULT

Context ID Register write. Fields:

FAULT unsigned int
: Fault status

PADDR unsigned int
: Physical Address (or 0 if unavailable)

ATOMIC_END_ACCESS

Bus trace access for atomics. Fields:

ACCESS_FAIL bool
: Memory access failed

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

LOAD_VALUE unsigned int
: Loaded values

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ATOMIC_START_ACCESS

Bus trace access for atomics. Fields:

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ArchMsg.Error.exit_code

Fields:

EXITCODE signed int
: Exit code to be returned

KIND string
: Component kind that invoked the exit code trace

REASON string
: Reason why the exit code trace was invoked (optional)

ArchMsg.Warning.TarmacReconstructor

Unexpected event happened during Tarmac reconstruction. Fields:

REASON string
: Reason

ArchMsg.Warning.Unpredictably Indexed PM Event Register

Fields:

IsDirect unsigned int
: Direct Access rather than indirect

N unsigned int
: Number of Accessible Registers

SEL unsigned int
: Selector

ArchMsg.Warning.branch_to_unaligned_address

Fields:

ADDR unsigned int
: Unaligned branch address

ArchMsg.Warning.cache_contents_unknown

A cache has been activated without being invalidated and may contain UNKNOWN content. Fields:

REGIME unsigned int
: Current translation regime

SIDE unsigned int
: 0 d-side cache or unified cache, 1 i-side

ArchMsg.Warning.dap_csw_bad_size

A write to CM3DAP CSW has an invalid size field. Fields:

DATA unsigned int
: bits[2:0] are size

ArchMsg.Warning.dcimvac_matches_watchpoint

Watchpoints matching an AArch32 DCIMVA are implementation defined.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_fieldmismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_fracbitsnegative

VCVT Instruction is unpredictable with a negative number of fraction bits.

ArchMsg.Warning.decode_initblock

Instruction is unpredictable in an IT block.

ArchMsg.Warning.decode_initblocknotlast

Instruction is unpredictable in an IT block when not the last.

ArchMsg.Warning.decode_invalid_condition

Instruction is unpredictable when the condition is not AL.

ArchMsg.Warning.decode_invalidfieldcombination

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_invalidvalue

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_mem_hint_unallocated

There is no memory hint allocated to this bit pattern.

ArchMsg.Warning.decode_registermatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registermismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registeroutofrange

Float point instruction is unpredictable indexing beyond the end of the register bank.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_sbzsbo

Reserved bits in the instruction are not canonical.

ArchMsg.Warning.decode_transactiontoobig

Attempt to perform a memory transaction of more than 128 bytes.

ArchMsg.Warning.decode_unpred_other

Instruction is unpredictable.

ArchMsg.Warning.decode_unpreduseofpc

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13inreglist

Instruction is unpredictable with R13 in register-list.

ArchMsg.Warning.decode_useofd16orhigher

Attempt to use register D16 or higher where it is not allowed.

ArchMsg.Warning.decode_writebackandbaseinlist

Instruction is unpredictable with write back when the base register is in the transfer list.

ArchMsg.Warning.decode_zeroregistersinlist

Instruction is unpredictable when no registers are to be transfered.

ArchMsg.Warning.recursive_branch

An instruction is performing a branch that targets the same instruction. If this was intended then a WFI might be more effective. Fields:

PC unsigned int
: Address of instruction causing the branch

ArchMsg.Warning.recursive_exception

An instruction has generated an exception that targets the same instruction. Fields:

CPSR unsigned int
: Processor state

PC unsigned int
: Address of instruction causing the exception

TYPE enum
: Type of exception

ArchMsg.Warning.reentrant_vector_catch

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR unsigned int
vector address

ArchMsg.Warning.reserved_it_state

Fields:

ITSTATE unsigned int
reserved IT state

ArchMsg.Warning.secure_vector_fetch_from_nonsecure

Secure vector table is being fetched from Non-secure memory .

ArchMsg.Warning.sg_in_it_block

Arm recommends that software does not place SG (security gate instruction) inside an IT instruction block.

ArchMsg.Warning.unpredictable_a32_breakpoint

Fields:

ADDR1 unsigned int
Address
ADDR2 unsigned int
Breakpoint address
BAS unsigned int
BAS Field
IS_ADDRESS_MISMATCH_BREAKPOINT bool
Is Mismatch breakpoint, not a match breakpoint
N unsigned int
Breakpoint number
OFFSET unsigned int
Indicates which halfword actually considering into the A32 instruction

ArchMsg.Warning.unpredictable_banked_register_access

Fields:

R unsigned int
R bit
SYSM unsigned int
SYSm

ArchMsg.Warning.unpredictable_ccfail_undef

Fields:

OPCODE unsigned int
opcode of the instruction
VADDR unsigned int
Virtual address of the instruction being executed

ArchMsg.Warning.unpredictable_ici_bits

ICI bits do not apply or are architecturally unpredictable for instruction.

ArchMsg.Warning.unpredictable_t32_breakpoint

Fields:

ADDR1 unsigned int
Address

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR2 unsigned int
: Breakpoint address

BAS unsigned int
: BAS Field

IS_ADDRESS_MISMATCH_BREAKPOINT bool
: Is Mismatch breakpoint, not a match breakpoint

N unsigned int
: Breakpoint number

OFFSET unsigned int
: Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_t32_vector_catch

Fields:

ADDR unsigned int
: Address

OFFSET unsigned int
: Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_watchpoint_far

Fields:

LOWER_BOUND unsigned int
: the lowest address accessed by the instruction that triggered the watchpoint

UPPER_BOUND unsigned int
: the highest watchpointed address accessed by that instruction

ArchMsg.Warning.warning_AdvSIMDExpandImmUnexpectedZero

AdvSIMDExpandImm may treat this immediate value as UNPREDICTABLE.

ArchMsg.Warning.warning_ConditionalSMC

SMC instruction has UNPREDICTABLE effects when conditional when combined with traps.

ArchMsg.Warning.warning_access_crosses_page_boundary

Access crosses page boundary.

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault_on_both_pages

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_spanning_different_memory

Fields:

ADDR unsigned int
: address of access crossing page

MEMTYPE_PAGE1 signed int
: First Page's memory type (0-Normal, 1-Non-Normal)

MEMTYPE_PAGE2 signed int
: Second Page's memory type (0-Normal, 1-Non-Normal)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_access_wraps_around_memory

Fields:

WIDTH unsigned int
: Address width

ArchMsg.Warning.warning_bx_from_thumbee

Fields:

ADDR unsigned int
: Destination address

ArchMsg.Warning.warning_ccsidr_unimplemented_level

Fields:

CSSELR unsigned int
: current effective value of CSSELR

ArchMsg.Warning.warning_change_to_ns_when_tge_set

Attempting to change NS to 1 when HCR.TGE = 1.

ArchMsg.Warning.warning_contiguous_bit_check_abort

Fields:

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ArchMsg.Warning.warning_contiguous_bit_error

Fields:

CONTIG_ADDR unsigned int
: address of the TLB first read, with which this entry is expected to be contiguous

CONTIG_DATA unsigned int
: expected contents based on the entry at CONTIG_ADDR

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ENTRY_DATA unsigned int
: contents of conflicting TLB entry

ArchMsg.Warning.warning_cp10_cp11_mismatch

Fields:

EL unsigned int
: EL owning the control

ArchMsg.Warning.warning_cp10_cp11_reserved_value

Fields:

CP10 unsigned int
: CP10 access

CP11 unsigned int
: CP11 access

ArchMsg.Warning.warning_csselr_level_out_of_range

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

IMPLEMENTED unsigned int
: number of cache levels implemented

LEVEL unsigned int
: written value of CSSELR.Level

TYPE enum
: The type of cache selected by CSSELR.InD and TnD

ArchMsg.Warning.warning_debug_flow_control_bits_not_obeyed

Fields:

ERROR enum
: type of error resulting

REGISTER enum
: register under consideration

ArchMsg.Warning.warning_debug_register_access_during_reset

Fields:

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

ArchMsg.Warning.warning_decode_cps_inconsistent_fields

The mode setting fields in the CPS instruction are inconsistent.

ArchMsg.Warning.warning_decode_invalid_state

Instruction is unpredictable in current exception-level/security state.

ArchMsg.Warning.warning_default_cacheable_mmu_on

Default cacheable is enabled (HCR.DC = 1) while MMU is enabled.

ArchMsg.Warning.warning_default_cacheable_vmmu_off

Default cacheable is enabled (HCR.DC = 1) while HCR.VM = 0.

ArchMsg.Warning.warning_deprecated_wvr_bit_2_set

Fields:

N unsigned int
: index

WVR unsigned int
: DBGWVR

ArchMsg.Warning.warning_eret_while_software_step_active_pending

Missing ISB between setting MDSCR_EL1.SS and ERET.

ArchMsg.Warning.warning_exclusive_to_non_normal

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_exclusive_to_non_writeback

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR unsigned int
Address

ArchMsg.Warning.warning_execute_from_device_memory

An attempt was made to execute from Device memory.

ArchMsg.Warning.warning_illegal_cpsr_mode

Fields:

NEW_MODE unsigned int
New value of CPSR.M

ArchMsg.Warning.warning_illegal_srs_mode

Fields:

MODE unsigned int
mode for Banked SP

ArchMsg.Warning.warning_implementation_defined_read_debug_register_in_SCS

Read Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

PPB_OFFSET unsigned int
debug register offset

ArchMsg.Warning.warning_implementation_defined_sequential_security_transitions_supported

The behaviour of a sequential instruction fetches that cross from non-secure to secure memory and contain SG instruction is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_implementation_defined_stack_limit_check_supported

It is IMPLEMENTATION DEFINED whether stack pointer limit checking is performed for the this instructions. Fields:

ADDRESS unsigned int
Address of instruction
OPCODE unsigned int
instruction opcode

ArchMsg.Warning.warning_implementation_defined_write_debug_register_in_SCS

Write Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

DATA unsigned int
data attempted to be written
PPB_OFFSET unsigned int
debug register offset

ArchMsg.Warning.warning_invalid_tcr_granule

Fields:

REQUEST signed int
page size requested (or 0 for reserved)
SUBSTITUTE signed int
best guess available page size
TG_ID bool
bits TG0 or TG1

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_load_multiple_user_registers_from_user_mode

An LDM(user registers) instruction executed from user mode.

ArchMsg.Warning.warning_load_pc_from_unaligned

PC loaded from an unaligned location.

ArchMsg.Warning.warning_mte_in_mem_attr_not_supported

Fields:

ATTR unsigned int
: Memory attribute

ArchMsg.Warning.warning_reserved_breakpoint_state_match

Fields:

IS_BREAKPOINT unsigned int
: Is breakpoint, not watchpoint

ArchMsg.Warning.warning_shareability

Unpredictable: combination of the 1st and 2nd stages of translation is Normal Inner Non-Cacheable, Outer Non-Cacheable.

ArchMsg.Warning.warning_software_step_set_while_enabled

MDSCR_EL1.SS set to 1 while software step debug exceptions are enabled.

ArchMsg.Warning.warning_thumb_instruction_wraps_around_memory

Thumb instruction warps around memory.

ArchMsg.Warning.warning_ttbr_sbz_bits_are_not_zero

Fields:

MSB unsigned int
: x-1

TTBR enum
: which TTBR register

ArchMsg.Warning.warning_unaligned_address_dbgdtrrx_write

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_unaligned_address_dbgdtrtx_read

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_unaligned_to_device

Fields:

ADDR unsigned int
: Address

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unaligned_to_strongly_ordered

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_unknown_sau_rnr

SAU_RNR was set to an unsupported value.

ArchMsg.Warning.warning_unpredictable_AIRCR_PRIP_and_BFHFNMINP

The effect of setting both AIRCR.BFHFNMINP and AIRCR.PRIP to 1 is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTCLRACTIVE_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTCLRACTIVE if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_and_SYSRESETREQ

When the processor is halted in Debug state, if a write to the register writes a 1 to both VECTRESET and SYSRESETREQ, the behavior is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTRESET if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_incorrect_VKEY

The value 0x05FA must be written to AIRCR.VKEY, otherwise the register write is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_EXE_RETURN_Reserved_Bit

EXC_RETURN[23:7] are reserved with the special condition that all bits should be written as one. Values other than all 1s are UNPREDICTABLE. Fields:

EXC_RETURN unsigned int
: Special PC value

SBO_MASK unsigned int
: Which bits Should Be One to not UNPRED

ArchMsg.Warning.warning_unpredictable_change_to_DEMCR_MON_STEP_at_insufficient_priority

The effect of changing DEMCR.MON_STEP at an execution priority that is lower than the priority of the DebugMonitor exception is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_change_to_priority_of_active_exception

Changing the priority of an active exception is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_clear_lspact

Fields:

DRVEXC_TAKEN unsigned int
: derived exception taken not pended

REACHED_LAST_STORE unsigned int
: only the last faulted

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_dbg_exit_unaligned_dlr

Debug exit to AArch32 with DLR[0] set to 1. PC[0] can be set to 0 or to DLR[0].

ArchMsg.Warning.warning_unpredictable_exception_catch

Generation of an exception catch event is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_inconsistent_state

state on exception return is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_instruction

exception return instruction is unpredictable.

ArchMsg.Warning.warning_unpredictable_in_debug_state

Instruction is UNPREDICTABLE when executed in debug state.

ArchMsg.Warning.warning_unpredictable_mair_encoding

Fields:

ATTR unsigned int
: Memory attribute

ArchMsg.Warning.warning_unpredictable_pmu_counter_access

Access to PMU counters from non-secure EL0 or EL1 is UNPREDICTABLE with MDCR_EL2.HPMN set to 0.

ArchMsg.Warning.warning_unpredictable_prioritization_breakpoint_match_vector_catch

Two events with the same priority occurred at the same instruction: Address Matching Vector Catch debug event, and Breakpoint debug event It is constrained unpredictable which is taken.

ArchMsg.Warning.warning_unpredictable_stack_selection

UNPREDICTABLE state on context switch.

ArchMsg.Warning.warning_unpredictable_tsize_out_of_range

Fields:

TCR enum
: Which TCR

TSIZE signed int
: Value of TSize

TSIZE_MAX signed int
: The maximum allowed value

TTBR unsigned int
: Which TTBR region

ArchMsg.Warning.warning_unpredictable_unaligned_pc_as_base_register

The PC value must be word-aligned, otherwise the behavior of the instruction is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_unaligned_pop_stack

Fields:

FRAME_ADDR unsigned int
: unaligned stack frame address

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_vmsa_memattrib

Fields:

TEXCB unsigned int
: texcb value

TEX_REMAP bool
: use tex remap

ArchMsg.Warning.warning_unpredictable_vtcr_t0sz_sl0

Fields:

SL0 signed int
: VTCR.SL0, starting level for stage 2 translation table walks

T0SZ signed int
: VTCR.T0SZ, required input address range

ArchMsg.Warning.warning_unpredictable_write_DHCSR

Fields:

NEW unsigned int
: data attempted to be written

OLD unsigned int
: previous value

ArchMsg.Warning.warning_unpredictable_write_SHCSR

UNPREDICTABLE setting a pending bit to 1 for an exception with priority >= execution priority.

ArchMsg.Warning.warning_unsupported_access_to_memory_mapped_register

Access to memory mapped register is unsupported.

ArchMsg.Warning.warning_user_jmcr_access

Fields:

WRITE unsigned int
: access is write

ArchMsg.Warning.warning_wcr_mask_and_bas

DBGWCR_EL1.MASK is non-zero and BAS is not 0xff.

ArchMsg.Warning.warning_wcr_mask_reserved

DBGWCR_ELn.MASK is set to a reserved value.

ArchMsg.Warning.warning_wcr_non_configuous_bas

DBGWCR_EL1.BAS has non-contiguous set of ones.

ArchMsg.Warning.warning_wvr_masked_bit_not_zero

DBGWVRn_EL1.VA contains a bit masked by DBGWCRn_EL1.MASK that is not zero.

ArchMsg.warning_unpred_system_register_access

Fields:

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

BRANCH_MISPREDICT

Simulating branch mispredict. Fields:

PC unsigned int
: Origin address (or 0 if unavailable)

BRA_DIR

Direct branches, to immediate address. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR

Indirect branches, perhaps to a register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR_CCFAIL

Indirect branches fail, perhaps to a register.

BRA_RET_CCFAIL

Branche return fail, perhaps to a register.

CACHE_MAINTENANCE_OP

Cache Maintenance Operation. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Specified MVA or Set/way

FUNCTION enum
: Clean / Invalidate

SCOPE enum
: Affected region

SIDE enum
: Inst / Data.

CCFAIL

Conditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCFAIL_UNC

Unconditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS

Conditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS_UNC

Unconditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CODE_CACHE_MAINT

Code cache maintenance. Fields:

END_INCL unsigned int
: Inclusive end addres for by VA or PA requests

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS bool
: Non-secure world for PA

START unsigned int
: Start address for by VA or PA requests

TYPE enum
: Request type.

COMPILE_BLOCK_END

Last instruction of basic block translated. Fields:

VADDR unsigned int
: Address of next instruction after this basic block

COMPILE_BLOCK_START

First instruction of basic block translated. Fields:

VADDR unsigned int
: Address of first instruction in basic block

COMPILE_INST

ARM instruction compiled. Fields:

DISASS string
: Disassembly of the instruction.

ISET enum
: The instruction set of this instruction.

ITSTATE unsigned int
: The ITSTATE current for the instruction.

OPCODE unsigned int
: The opcode of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

CONTEXTIDR

Context ID Register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

NS bool
: Secure or nonsecure banked register is accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

CONTEXT_SYNC

Called for every context synchronization event. Fields:

ARCH_CSE bool
: Architectural Context Synchronization Event

SYSREG_SYNC bool
: System register synchronization event

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_ENDIAN

Core BE8 Big-Endian state changed. Fields:

BE8 bool
: Core BE8 Big-Endian state

CORE_INFO

Static processor attributes. Only triggered by a call to DumpState(). Fields:

ARCH_PROFILE enum
: The architecture profile of the core.

CLUSTER_ID unsigned int
: The cluster ID of this processor.

CORE_NUM unsigned int
: The number of this core in an MP processor.

DETAILED_CORE_FEATURES enum
: Detailed core features as a bitfield. Enum values are bitmasks to extract the info from the field

FPU_VERSION enum
: The VFP version implemented by the core.

MEM_ARCH enum
: The memory architecture of the core.

MIDR_EL1 unsigned int
: MIDR_EL1 value

NUM_CORES unsigned int
: The number of cores in this MP processor.

QUANTUM_SIZE unsigned int
: The default quantum size of the core.

SECURITY_FEATURES bool
: Does the core have security features?

CORE_LOADS

Processor load accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CORE_REGS

Changes of the core registers R0 to R14. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number, 0 to 14.

OLD_VALUE unsigned int
: The old value overwritten.

PHYS_ID enum
: The physical register accessed.

VALUE unsigned int
: The new value written to the register.

CORE_STORES

Processor store accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CRYPTO_SPEC

Every crypto instruction speculatively executed.

DATA_CACHE_SET_ALLOC_TAGS

Set allocation tags from DC GVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of memory in bytes.

TAG unsigned int
: The value of the tag being set.

VADDR unsigned int
: Virtual address used for setting allocation tags.

DATA_CACHE_ZERO

Zero and invalidate cache line from DC ZVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of zero'd memory in bytes.

VADDR unsigned int
: Virtual address of the zero'd cache line.

DEBUG_EVENT

Hardware debug support event. Fields:

EVENT enum
: Description of event

VALUE unsigned int
: data value

DTCM_ACCESS

Trace TCM accesses.

DTCM_ACCESS_RAZ_WI

Trace TCM accesses. Fields:

ADDR unsigned int
: Address of the RAZ/WI access.

DWT_MATCH

DWT comparator matches. Fields:

NUM unsigned int
: DWT comparator number

TYPE enum
: DWT comparator configuration

END_COMPILE

Compilation end. Fields:

END_OF_PAGE_COUNT unsigned int
: Number of basic blocks exited due to page boundary since START_COMPILE.

FETCHFAIL_COUNT unsigned int
: Number of basic blocks exited due to ifetch failure START_COMPILE.

INST_COUNT unsigned int
: Number of instructions compiled since START_COMPILE.

NONSEQ_COUNT unsigned int
: Number of basic blocks exited due to non-sequential instructions since START_COMPILE.

PAGE_STRADDLE_COUNT unsigned int
: Number of basic blocks exited due to unaligned instructions crossing page since START_COMPILE.

EXCEPTION

Exceptions which are taken. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: The instruction set of the processor when the exception occurred.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LR unsigned int
: The value assigned to the link register.

NS enum
: The core's non-secure bit.

PC unsigned int
: The location where the exception occurred.

TARGET_ISET enum
: The instruction set of the exception handler code.

TARGET_PC unsigned int
: The address the exception branches to.

VECTOR enum
: The exception vector.

EXCEPTION_END

Every exception completed. Fields:

NS enum
: The core's non-secure bit.

EXCEPTION_ENTRY

Event marking the entry of an exception. All pushes of register to the stack will follow this event. Fields:

PC unsigned int
: The location where the exception occurred.

VECTOR enum
: The exception that occurred.

EXCEPTION_RAISE

Every exception raised.

EXCEPTION_RETURN

Marks the end of an exception, but on an M core does not indicate a program flow change (branch). In most situations it will be followed by an indirect branch. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

PC unsigned int
: The address of the branch instruction.

EXCEPTION_RETURN_PREBRANCH

Exception return event that must occur before PC and processor state have updated.

EXCEPTION_START

Every exception started. Fields:

NS enum
: The core's non-secure bit.

EXCEPTION_VECTOR_FETCH

I-side vector fetch for M-class exception. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Result of fetch if OK

RESPONSE enum
: 0=Aborted, 1=OK

VADDR unsigned int
: Where the vector is fetched from (including VTOR)

FP_STATE

Floating point state. Fields:

FPSR unsigned int
: FPSR register value.

MASK unsigned int
: The mask indicating updated cumulative bits.

HLT

HLT instruction occurred. Fields:

IMM16 unsigned int
: The 16-bit immediate value encoded in HLT instruction.

INFO_EXCEPTION_REASON

Provide information of real cause of exception. Fields:

FaultCause enum
: Which bit of which V7M/V8M-mainline FSR/HFSR would have been set.

PC unsigned int
: The location where the exception occurred.

PHASE enum
: What we are doing with the exception.

REASONS enum
: Why we are doing it.

SecurityState enum
: Privacy State of exception

VECTOR enum
: The exception that occurred.

INFO_FP_CONTEXT

Inform when ExecuteFPCheck writes CONTROL.FPCA and FPSCR etc. Fields:

WHAT enum
: What we are doing.

INFO_STACKING

Provide information of real cause of exception. Fields:

PHASE enum
: Are we starting or ending a sequence of stack writes.

REASONS enum
: Why we are doing it.

INST

Every instruction executed. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DEBUG_STATE bool
: The instruction is executed in debug state.

DISASS string
: Disassembly of instruction.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The current instruction set.

ITSTATE unsigned int
: The current ITSTATE.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

MODE enum
: The mode the core is in.

NS unsigned int
: The core's non-secure bit.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

OPCODE unsigned int
: The opcode of the instruction.

PADDR unsigned int
: The physical address of the instruction.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

INST_END

Every instruction completed.

INST_START

Every instruction started. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

INST_STRADDLE

Instruction straddles boundary. Fields:

ISET enum
: The current instruction set.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

ISB

ISB instruction occurred.

ITCM_ACCESS

Trace TCM accesses.

ITCM_ACCESS_RAZ_WI

Trace TCM accesses. Fields:

ADDR unsigned int
: Address of the RAZ/WI access.

LOAD_MULTIPLE_REGS_LIST

List of destination registers (not source) of a multiple registers load instruction. Fields:

REG string
: Register to add into the list

LOCAL_MONITOR

Local monitor activity. Fields:

PADDR unsigned int
: Local Monitor Address

State enum
: State of the monitor (Open/Exclusive)

LOCKUP_CYCLE

This event is triggered if the core enters or stays in the lock-up state. Fields:

CAUSE enum
: What causes the lockup?.

LOCKUP_ENTRY

This event is triggered if the core enters the lock-up state. Fields:

PC unsigned int
: The PC when the lockup state is entered.

LOCKUP_EXIT

This event is triggered if the core leaves the lock-up state.

MODE_CHANGE

Mode change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: The new mode.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OLD_MODE enum
: The old mode.

MPU_TRANS

Address translation information. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

EXEC_PERM enum
: Execution Permission

IDAU_REG_NUM enum
: IDAU region-number or code

INNERCACHE_RA bool
: Is the inner cache allocate on read

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

MEMTYPE enum
: Memory type.

NSDESC enum
: Security-attributes of the transaction.

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

PADDR unsigned int
: Address of the access.

PAGESIZE unsigned int
: Page size as log2(size).

READ_PERM enum
: Read Permission

REG_NUM signed int
: MPU region-number 0-15 (else 0xFF=none, 0xBB/0xDD=background/default)

SAU_REG_NUM enum
: SAU region-number or code

SAU_RGN_TYPE enum
: Privacy Region Type

SH enum
: Shareability

SIDE enum
: Inst / Data.

WRITE_PERM enum
: Write Permission

PERIODIC

Called for every quantum. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

PC unsigned int
: The address of the next instruction to be executed on this CPU.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PPB_READ_ACCESS

Trace reads from the core to memory mapped registers in the PPB (private peripheral bus) address range. Fields:

ACCESS_SIZE unsigned int
: Logarithm of the access size: 0=byte, 1=halfword, 2=word, 3=doubleword, ...

BURST_LEN unsigned int
: The number of data transfers (beats) in this burst.

DATA unsigned int
: The data transferred.

INNER_ATTRIBUTES unsigned int
: The inner attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

INST bool
: Is an instruction fetch.

LOCK enum
: Normal / Locked / Exclusive.

MASTER_ID unsigned int
: The AXI master ID.

NS bool
: Is a nonsecure access.

OUTER_ATTRIBUTES unsigned int
: The outer attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

PADDR unsigned int
: Physical address of access.

PRIV bool
: Is privileged access.

RESPONSE enum
: Whether the transaction was successful, or an error occurred.

USER_FLAGS unsigned int
: Core specific additional signals.

PPB_WRITE_ACCESS

Trace writes from the core to memory mapped registers in the PPB (private peripheral bus) address range. Fields:

ACCESS_SIZE unsigned int
: Logarithm of the access size: 0=byte, 1=halfword, 2=word, 3=doubleword, ...

BURST_LEN unsigned int
: The number of data transfers (beats) in this burst.

DATA unsigned int
: The data transferred.

INNER_ATTRIBUTES unsigned int
: The inner attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

INST bool
: Is an instruction fetch.

LOCK enum
: Normal / Locked / Exclusive.

MASTER_ID unsigned int
: The AXI master ID.

NS bool
: Is a nonsecure access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OUTER_ATTRIBUTES unsigned int
: The outer attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

PADDR unsigned int
: Physical address of access.

PRIV bool
: Is privileged access.

RESPONSE enum
: Whether the transaction was successful, or an error occurred.

USER_FLAGS unsigned int
: Core specific additional signals.

PRELOAD_DATA

Data preload from PLD instruction. Fields:

VADDR unsigned int
: Virtual address of the data that should be preloaded.

PRELOAD_INST

Instruction preload from PLI instruction. Fields:

VADDR unsigned int
: Virtual address of the instruction that should be preloaded.

PRE_CORE_LOAD

Trace just before a core load. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_CORE_STORE

Trace just before a core store. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PSTATE_PACTIVE_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

PSTATE_PACTIVE_OPMODE_TRANSITION_DENIED

Trace the denied pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OLD_STATE enum
: Operating mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_POWER_TRANSITION

Trace pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

PSTATE_PACTIVE_POWER_TRANSITION_DENIED

Trace the denied pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_THREAD_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions for threads on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

RUN_STATE

Run state transition. Fields:

INST_COUNT unsigned int
: Ticks count at point of transition.

NEW enum
: New run state.

OLD enum
: Old run state.

SEMIHOSTING_CALL

Call of a semihost function occurred. Fields:

PC unsigned int
: The program counter after the semihosting call.

SEMIHOSTING_PRECALL

About to call semihost. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

EL2 bool
: Use EL2 translation regime.

NS bool
: Is Non-Secure.

REG_WIDTH unsigned int
: The current register width in bytes.

SIGNAL

External signal state change. Fields:

SIGNAL enum
: Signal that changed

STATE bool
: Signal asserted state

START_COMPILE

Compilation started. Fields:

VADDR unsigned int
: Instruction where compilation begins.

STG_MULTIPLE_MICRO_OPS_ADDRESS_RANGE

Address range for MTE stores with multiple uops. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

STORE_MULTIPLE_REGS_ADDRESS_RANGE

Address range for stores with multiple registers. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

SUBOPTIMAL_LDST_RETIRED

suboptimal load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1: half-word, 2: word ...)

OPTIMAL_ALIGN_SELECTOR unsigned int
: Selector to decide the optimal alignment size (1: access width, 2: 4-bytes, 3: 8-bytes, ..., 12: 4-kbytes)

VADDR unsigned int
: The virtual address of the access

SYNC

Called for every synchronization, that is every quantum break. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

LOCAL_QUANTUM unsigned int
: The local quantum of this CPU.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOCAL_TIME unsigned int
: The local time of this CPU.

SYNC_REASON enum
: Reason for sync.

SYSCALL

System call instruction executed. Fields:

IMM unsigned int
: Immediate value of the system call instruction.

TYPE enum
: System call type.

VADDR unsigned int
: Instruction that caused the system call.

SYSREG_READ

Trace reads from system registers. Fields:

REG_NAME enum
: Register number

VALUE unsigned int
: The value read.

SYSREG_UPDATE32

Triggers when the system updates a register. Fields:

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

SYSREG_UPDATE64

Triggers when the system updates a register. Fields:

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

SYSREG_WRITE

Trace write to system registers. Fields:

REG_ACCESS enum
: Register access status

REG_NAME enum
: Register number

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: Value written to the register.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

UNALIGNED_LDST_RETIRED

Processor unaligned load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1=>half-word, 2->word ...)

NUMBER_OF_BEATS unsigned int
: Number of accesses (beats) in this burst (Total size in bytes is ACCESS_SIZE * NUMBER_OF_BEATS)

VADDR unsigned int
: The virtual address of the access

VFP_D_REGS

VFP/NEON D 64 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_Q_REGS

VFP/NEON Q 128 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding byte in the register. For example, 0x1 would indicate a write to VALUE[7:0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_SYS_REGS

Writes to the VFP/NEON units system registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: Which VFP system register is written.

MASK unsigned int
: The mask indicating updated cumulative bits.

OLD_VALUE unsigned int
: The register's old value overwritten.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VALUE unsigned int
: The new value written to the VFP system register.

VFP_S_REGS

VFP/NEON S 32 bit register write. Fields:

ALIASING enum
: Type of register aliasing used.

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

WAYPOINT

Signals end of basic block execution. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: Origin instructions set

IS_COND bool
: Indicates if this is a conditional waypoint.

PC unsigned int
: Origin address (or 0 if unavailable)

TAKEN bool
: Indicates if this waypoint was taken.

TARGET unsigned int
: Destination address

TARGET_ISET enum
: Destination instructions set

WFE_END

WFE ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFE.

WFE_EVENT_REGISTER

WFE event register status: set/clear, reason. Fields:

INST_COUNT unsigned int
: Ticks count.

REASON enum
: Reason for set/clear. Only REASON==0 (Cleared by WFE) clears the bit, all other reasons set it.

WFE_IGNORED

WFE ignored. Fields:

EVENT bool
: This WFE was ignored because the event register was set.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: Ticks count when ignoring WFE.

TRAPPED bool
: This WFE was trapped.

WFE_START

WFE entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFE.

WFI_END

WFI ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFI.

WFI_IGNORED

WFI ignored. Fields:

DISABLED bool
: This WFI was ignored because WFI is disabled.

INST_COUNT unsigned int
: Ticks count when ignoring WFI.

REASON enum
: specifies reason why WFI trace was ignored

TRAPPED bool
: This WFI was trapped.

WFI_START

WFI entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFI.

WFI_WAKEUP

WFI wakeup. Fields:

INST_COUNT unsigned int
: Ticks count when WFI wakeup occurred.

REASON enum
: Reason for wakeup.

XPSR

Changes to the xPSR register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

OLD_VALUE unsigned int
: The old xPSR value

UNKNOWN unsigned int
: Bits within the register that have unknown value.

VALUE unsigned int
: The new xPSR value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.6 ARM_Cortex-A15 - trace

This section describes the trace sources.

ASYNC_MEMORY_FAULT

Context ID Register write. Fields:

FAULT unsigned int
: Fault status in ESR format

PADDR unsigned int
: Physical Address (or 0 if unavailable)

VADDR signed int
: Virtual Address (or 0 if unavailable)

ATOMIC_END_ACCESS

Bus trace access for atomics. Fields:

ACCESS_FAIL bool
: Memory access failed

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

LOAD_VALUE unsigned int
: Loaded values

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ATOMIC_START_ACCESS

Bus trace access for atomics. Fields:

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

NSDESC unsigned int
: The security state of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ArchMsg.Error.exit_code

Fields:

EXITCODE signed int
: Exit code to be returned

KIND string
: Component kind that invoked the exit code trace

REASON string
: Reason why the exit code trace was invoked (optional)

ArchMsg.Warning.IgnoredWarmResetRequest

Event fires when software attempts to request warm reset through write to deprecated DBGPRCR.CWRR.

ArchMsg.Warning.TarmacReconstructor

Unexpected event happened during Tarmac reconstruction. Fields:

REASON string
: Reason

ArchMsg.Warning.Unpredictably Indexed PM Event Register

Fields:

IsDirect unsigned int
: Direct Access rather than indirect

N unsigned int
: Number of Accessible Registers

SEL unsigned int
: Selector

ArchMsg.Warning.branch_to_unaligned_address

Fields:

ADDR unsigned int
: Unaligned branch address

ArchMsg.Warning.cache_contents_unknown

A cache has been activated without being invalidated and may contain UNKNOWN content. Fields:

REGIME unsigned int
: Current translation regime

SIDE unsigned int
: 0 d-side cache or unified cache, 1 i-side

ArchMsg.Warning.dcimvac_matches_watchpoint

Watchpoints matching an AArch32 DCIMVA are implementation defined.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_fieldmismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_fracbitsnegative

VCVT Instruction is unpredictable with a negative number of fraction bits.

ArchMsg.Warning.decode_initblock

Instruction is unpredictable in an IT block.

ArchMsg.Warning.decode_initblocknotlast

Instruction is unpredictable in an IT block when not the last.

ArchMsg.Warning.decode_invalid_condition

Instruction is unpredictable when the condition is not AL.

ArchMsg.Warning.decode_invalidfieldcombination

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_invalidvalue

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_mem_hint_unallocated

There is no memory hint allocated to this bit pattern.

ArchMsg.Warning.decode_registermatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registermismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registeroutofrange

Float point instruction is unpredictable indexing beyond the end of the register bank.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_sbzsbo

Reserved bits in the instruction are not canonical.

ArchMsg.Warning.decode_transactiontoobig

Attempt to perform a memory transaction of more than 128 bytes.

ArchMsg.Warning.decode_unpred_other

Instruction is unpredictable.

ArchMsg.Warning.decode_unpreduseofpc

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13inreglist

Instruction is unpredictable with R13 in register-list.

ArchMsg.Warning.decode_useofd16orhigher

Attempt to use register D16 or higher where it is not allowed.

ArchMsg.Warning.decode_writebackandbaseinlist

Instruction is unpredictable with write back when the base register is in the transfer list.

ArchMsg.Warning.decode_zeroregistersinlist

Instruction is unpredictable when no registers are to be transfered.

ArchMsg.Warning.recursive_branch

An instruction is performing a branch that targets the same instruction. If this was intended then a WFI might be more effective. Fields:

PC unsigned int
: Address of instruction causing the branch

ArchMsg.Warning.recursive_exception

An instruction has generated an exception that targets the same instruction. Fields:

CPSR unsigned int
: Processor state

PC unsigned int
: Address of instruction causing the exception

TYPE enum
: Type of exception

ArchMsg.Warning.reentrant_vector_catch

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR unsigned int
vector address

ArchMsg.Warning.reserved_it_state

Fields:

ITSTATE unsigned int
reserved IT state

ArchMsg.Warning.tlb_contents_unknown

Translation has been enabled without invalidating the TLB. Fields:

EL unsigned int
Exception level of translation regime
INVALIDITY unsigned int
Which TLB may be invalid (if TLBs are separate)
NON_SECURE unsigned int
Security state of translation regime
VMID unsigned int
Current VMID, if applicable

ArchMsg.Warning.unpredictable_a32_breakpoint

Fields:

ADDR1 unsigned int
Address
ADDR2 unsigned int
Breakpoint address
BAS unsigned int
BAS Field
IS_ADDRESS_MISMATCH_BREAKPOINT bool
Is Mismatch breakpoint, not a match breakpoint
N unsigned int
Breakpoint number
OFFSET unsigned int
Indicates which halfword actually considering into the A32 instruction

ArchMsg.Warning.unpredictable_banked_register_access

Fields:

R unsigned int
R bit
SYSM unsigned int
SYSm

ArchMsg.Warning.unpredictable_ccfail_undef

Fields:

OPCODE unsigned int
opcode of the instruction
VADDR unsigned int
Virtual address of the instruction being executed

ArchMsg.Warning.unpredictable_ici_bits

ICI bits do not apply or are architecturally unpredictable for instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.unpredictable_t32_breakpoint

Fields:

ADDR1 unsigned int
: Address

ADDR2 unsigned int
: Breakpoint address

BAS unsigned int
: BAS Field

IS_ADDRESS_MISMATCH_BREAKPOINT bool
: Is Mismatch breakpoint, not a match breakpoint

N unsigned int
: Breakpoint number

OFFSET unsigned int
: Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_t32_vector_catch

Fields:

ADDR unsigned int
: Address

OFFSET unsigned int
: Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_watchpoint_far

Fields:

LOWER_BOUND unsigned int
: the lowest address accessed by the instruction that triggered the watchpoint

UPPER_BOUND unsigned int
: the highest watchpointed address accessed by that instruction

ArchMsg.Warning.warning_AdvSIMDExpandImmUnexpectedZero

AdvSIMDExpandImm may treat this immediate value as UNPREDICTABLE.

ArchMsg.Warning.warning_ConditionalSMC

SMC instruction has UNPREDICTABLE effects when conditional when combined with traps.

ArchMsg.Warning.warning_access_crosses_page_boundary

Access crosses page boundary.

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault_on_both_pages

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_spanning_different_memory

Fields:

ADDR unsigned int
: address of access crossing page

MEMTYPE_PAGE1 signed int
: First Page's memory type (0-Normal, 1-Non-Normal)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MEMTYPE_PAGE2 signed int
Second Page's memory type (0-Normal, 1-Non-Normal)

ArchMsg.Warning.warning_access_wraps_around_memory

Fields:

WIDTH unsigned int
Address width

ArchMsg.Warning.warning_bx_from_thumbee

Fields:

ADDR unsigned int
Destination address

ArchMsg.Warning.warning_ccsidr_unimplemented_level

Fields:

CSSELR unsigned int
current effective value of CSSELR

ArchMsg.Warning.warning_change_to_ns_when_tge_set

Attempting to change NS to 1 when HCR.TGE = 1.

ArchMsg.Warning.warning_contiguous_bit_check_abort

Fields:

ENTRY_ADDR unsigned int
address of conflicting TLB entry

ArchMsg.Warning.warning_contiguous_bit_error

Fields:

CONTIG_ADDR unsigned int
address of the TLB first read, with which this entry is expected to be contiguous
CONTIG_DATA unsigned int
expected contents based on the entry at CONTIG_ADDR
ENTRY_ADDR unsigned int
address of conflicting TLB entry
ENTRY_DATA unsigned int
contents of conflicting TLB entry

ArchMsg.Warning.warning_cp10_cp11_mismatch

Fields:

EL unsigned int
EL owning the control

ArchMsg.Warning.warning_cp10_cp11_reserved_value

Fields:

CP10 unsigned int
CP10 access
CP11 unsigned int
CP11 access

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_csselr_level_out_of_range

Fields:

IMPLEMENTED unsigned int
: number of cache levels implemented

LEVEL unsigned int
: written value of CSSELR.Level

TYPE enum
: The type of cache selected by CSSELR.InD and TnD

ArchMsg.Warning.warning_debug_flow_control_bits_not_obeyed

Fields:

ERROR enum
: type of error resulting

REGISTER enum
: register under consideration

ArchMsg.Warning.warning_debug_register_access_during_reset

Fields:

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

ArchMsg.Warning.warning_decode_cps_inconsistent_fields

The mode setting fields in the CPS instruction are inconsistent.

ArchMsg.Warning.warning_decode_invalid_state

Instruction is unpredictable in current exception-level/security state.

ArchMsg.Warning.warning_default_cacheable_mmu_on

Default cacheable is enabled (HCR.DC = 1) while MMU is enabled.

ArchMsg.Warning.warning_default_cacheable_vmmu_off

Default cacheable is enabled (HCR.DC = 1) while HCR.VM = 0.

ArchMsg.Warning.warning_deprecated_wvr_bit_2_set

Fields:

N unsigned int
: index

WVR unsigned int
: DBGWVR

ArchMsg.Warning.warning_eret_while_software_step_active_pending

Missing ISB between setting MDSCR_EL1.SS and ERET.

ArchMsg.Warning.warning_exclusive_to_non_normal

Fields:

ADDR unsigned int
: Address

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_exclusive_to_non_writeback

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_execute_from_device_memory

An attempt was made to execute from Device memory.

ArchMsg.Warning.warning_illegal_cpsr_mode

Fields:

NEW_MODE unsigned int
: New value of CPSR.M

ArchMsg.Warning.warning_illegal_srs_mode

Fields:

MODE unsigned int
: mode for Banked SP

ArchMsg.Warning.warning_implementation_defined_read_debug_register_in_SCS

Read Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_implementation_defined_sequential_security_transitions_supported

The behaviour of a sequential instruction fetches that cross from non-secure to secure memory and contain SG instruction is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_implementation_defined_stack_limit_check_supported

It is IMPLEMENTATION DEFINED whether stack pointer limit checking is performed for the this instructions. Fields:

ADDRESS unsigned int
: Address of instruction

OPCODE unsigned int
: instruction opcode

ArchMsg.Warning.warning_implementation_defined_write_debug_register_in_SCS

Write Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

DATA unsigned int
: data attempted to be written

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_invalid_tcr_granule

Fields:

REQUEST signed int
: page size requested (or 0 for reserved)

SUBSTITUTE signed int
: best guess available page size

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

TG_ID bool
bits TG0 or TG1

ArchMsg.Warning.warning_load_multiple_user_registers_from_user_mode

An LDM(user registers) instruction executed from user mode.

ArchMsg.Warning.warning_load_pc_from_unaligned

PC loaded from an unaligned location.

ArchMsg.Warning.warning_mte_in_mem_attr_not_supported

Fields:

ATTR unsigned int
Memory attribute

ArchMsg.Warning.warning_reserved_breakpoint_state_match

Fields:

IS_BREAKPOINT unsigned int
Is breakpoint, not watchpoint

ArchMsg.Warning.warning_shareability

Unpredictable: combination of the 1st and 2nd stages of translation is Normal Inner Non-Cacheable, Outer Non-Cacheable.

ArchMsg.Warning.warning_software_step_set_while_enabled

MDSCR_EL1.SS set to 1 while software step debug exceptions are enabled.

ArchMsg.Warning.warning_thumb_instruction_wraps_around_memory

Thumb instruction warps around memory.

ArchMsg.Warning.warning_ttbr_sbz_bits_are_not_zero

Fields:

MSB unsigned int
x-1
TTBR enum
which TTBR register

ArchMsg.Warning.warning_unaligned_address_dbgdtrrx_write

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_address_dbgdtrtx_read

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_device

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_strongly_ordered

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unknown_sau_rnr

SAU_RNR was set to an unsupported value.

ArchMsg.Warning.warning_unpredictable_AIRCR_PRIP_and_BFHFNMINP

The effect of setting both AIRCR.BFHFNMINP and AIRCR.PRIP to 1 is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTCLRACTIVE_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTCLRACTIVE if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_and_SYSRESETREQ

When the processor is halted in Debug state, if a write to the register writes a 1 to both VECTRESET and SYSRESETREQ, the behavior is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTRESET if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_incorrect_VKEY

The value 0x05FA must be written to AIRCR.VKEY, otherwise the register write is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_EXE_RETURN_Reserved_Bit

EXC_RETURN[23:7] are reserved with the special condition that all bits should be written as one. Values other than all 1s are UNPREDICTABLE. Fields:

EXC_RETURN unsigned int
Special PC value
SBO_MASK unsigned int
Which bits Should Be One to not UNPRED

ArchMsg.Warning.warning_unpredictable_change_to_DEMCR_MON_STEP_at_insufficient_priority

The effect of changing DEMCR.MON_STEP at an execution priority that is lower than the priority of the DebugMonitor exception is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_change_to_priority_of_active_exception

Changing the priority of an active exception is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_clear_lspact

Fields:

DRVEXC_TAKEN unsigned int
derived exception taken not pended

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

REACHED_LAST_STORE unsigned int
only the last faulted

ArchMsg.Warning.warning_unpredictable_dbg_exit_unaligned_dlr

Debug exit to AArch32 with DLR[0] set to 1. PC[0] can be set to 0 or to DLR[0].

ArchMsg.Warning.warning_unpredictable_exception_catch

Generation of an exception catch event is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_inconsistent_state

state on exception return is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_instruction

exception return instruction is unpredictable.

ArchMsg.Warning.warning_unpredictable_in_debug_state

Instruction is UNPREDICTABLE when executed in debug state.

ArchMsg.Warning.warning_unpredictable_mair_encoding

Fields:

ATTR unsigned int
Memory attribute

ArchMsg.Warning.warning_unpredictable_pmu_counter_access

Access to PMU counters from non-secure EL0 or EL1 is UNPREDICTABLE with MDCR_EL2.HPMN set to 0.

ArchMsg.Warning.warning_unpredictable_prioritization_breakpoint_match_vector_catch

Two events with the same priority occurred at the same instruction: Address Matching Vector Catch debug event, and Breakpoint debug event It is constrained unpredictable which is taken.

ArchMsg.Warning.warning_unpredictable_stack_selection

UNPREDICTABLE state on context switch.

ArchMsg.Warning.warning_unpredictable_tsize_out_of_range

Fields:

TCR enum
Which TCR
TSIZE signed int
Value of TSize
TSIZE_MAX signed int
The maximum allowed value
TTBR unsigned int
Which TTBR region

ArchMsg.Warning.warning_unpredictable_unaligned_pc_as_base_register

The PC value must be word-aligned, otherwise the behavior of the instruction is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_unaligned_pop_stack

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

FRAME_ADDR unsigned int
: unaligned stack frame address

ArchMsg.Warning.warning_unpredictable_vmsa_memattrib

Fields:

TEXCB unsigned int
: texcb value

TEX_REMAP bool
: use tex remap

ArchMsg.Warning.warning_unpredictable_vtcr_t0sz_sl0

Fields:

SL0 signed int
: VTCR.SL0, starting level for stage 2 translation table walks

T0SZ signed int
: VTCR.T0SZ, required input address range

ArchMsg.Warning.warning_unpredictable_write_DHCSR

Fields:

NEW unsigned int
: data attempted to be written

OLD unsigned int
: previous value

ArchMsg.Warning.warning_unpredictable_write_SHCSR

UNPREDICTABLE setting a pending bit to 1 for an exception with priority >= execution priority.

ArchMsg.Warning.warning_unsupported_access_to_memory_mapped_register

Access to memory mapped register is unsupported.

ArchMsg.Warning.warning_user_jmcr_access

Fields:

WRITE unsigned int
: access is write

ArchMsg.Warning.warning_wcr_mask_and_bas

DBGWCR_EL1.MASK is non-zero and BAS is not 0xff.

ArchMsg.Warning.warning_wcr_mask_reserved

DBGWCR_ELn.MASK is set to a reserved value.

ArchMsg.Warning.warning_wcr_non_configuous_bas

DBGWCR_EL1.BAS has non-contiguous set of ones.

ArchMsg.Warning.warning_wvr_masked_bit_not_zero

DBGWVRn_EL1.VA contains a bit masked by DBGWCRn_EL1.MASK that is not zero.

BRANCH_MISPREDICT

Simulating branch mispredict. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PC unsigned int
: Origin address (or 0 if unavailable)

BRA_DIR

Direct branches, to immediate address. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR

Indirect branches, perhaps to a register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR_CCFAIL

Indirect branches fail, perhaps to a register.

BRA_RET_CCFAIL

Branche return fail, perhaps to a register.

CACHE_MAINTENANCE_OP

Cache Maintenance Operation. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Specified MVA or Set/way

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

FUNCTION enum
: Clean / Invalidate

SCOPE enum
: Affected region

SIDE enum
: Inst / Data.

CCFAIL

Conditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCFAIL_UNC

Unconditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS

Conditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS_UNC

Unconditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CODE_CACHE_MAINT

Code cache maintenance. Fields:

END_INCL unsigned int
: Inclusive end addres for by VA or PA requests

NS bool
: Non-secure world for PA

START unsigned int
: Start address for by VA or PA requests

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

TYPE enum
: Request type.

COMPILE_BLOCK_END

Last instruction of basic block translated. Fields:

VADDR unsigned int
: Address of next instruction after this basic block

COMPILE_BLOCK_START

First instruction of basic block translated. Fields:

VADDR unsigned int
: Address of first instruction in basic block

COMPILE_INST

ARM instruction compiled. Fields:

DISASS string
: Disassembly of the instruction.

ISET enum
: The instruction set of this instruction.

ITSTATE unsigned int
: The ITSTATE current for the instruction.

OPCODE unsigned int
: The opcode of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

CONTEXTIDR

Context ID Register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

NS bool
: Secure or nonsecure banked register is accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

CONTEXT_SYNC

Called for every context synchronization event. Fields:

ARCH_CSE bool
: Architectural Context Synchronization Event

SYSREG_SYNC bool
: System register synchronization event

CORE_ENDIAN

Core BE8 Big-Endian state changed. Fields:

BE8 bool
: Core BE8 Big-Endian state

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_INFO

Static processor attributes. Only triggered by a call to DumpState(). Fields:

ARCH_PROFILE enum
: The architecture profile of the core.

CLUSTER_ID unsigned int
: The cluster ID of this processor.

CORE_NUM unsigned int
: The number of this core in an MP processor.

DETAILED_CORE_FEATURES enum
: Detailed core features as a bitfield. Enum values are bitmasks to extract the info from the field

FPU_VERSION enum
: The VFP version implemented by the core.

MEM_ARCH enum
: The memory architecture of the core.

MIDR_EL1 unsigned int
: MIDR_EL1 value

NUM_CORES unsigned int
: The number of cores in this MP processor.

QUANTUM_SIZE unsigned int
: The default quantum size of the core.

SECURITY_FEATURES bool
: Does the core have security features?

CORE_LOADS

Processor load accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CORE_REGS

Changes of the core registers R0 to R14. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number, 0 to 14.

MODE enum
: Bank of the register accessed.

OLD_VALUE unsigned int
: The old value overwritten.

PHYS_ID enum
: The physical register accessed.

VALUE unsigned int
: The new value written to the register.

CORE_STORES

Processor store accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CP14_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP14_WRITE

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

CP15_WRITE

System Control Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

CPSR

CPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

OLD_VALUE unsigned int
: The old CPSR value

UNKNOWN unsigned int
: Bits within the register that have unknown value.

VALUE unsigned int
: The new CPSR value

CRYPTO_SPEC

Every crypto instruction speculatively executed.

CorePowerStateChange

Core power-up state has changed. Fields:

PU bool
: Whether the core powered up

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DATA_CACHE_SET_ALLOC_TAGS

Set allocation tags from DC GVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of memory in bytes.

TAG unsigned int
: The value of the tag being set.

VADDR unsigned int
: Virtual address used for setting allocation tags.

DATA_CACHE_ZERO

Zero and invalidate cache line from DC ZVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of zero'd memory in bytes.

VADDR unsigned int
: Virtual address of the zero'd cache line.

DEBUG_EVENT

Hardware debug support event. Fields:

EVENT enum
: Description of event

VALUE unsigned int
: data value

END_COMPILE

Compilation end. Fields:

END_OF_PAGE_COUNT unsigned int
: Number of basic blocks exited due to page boundary since START_COMPILE.

FETCHFAIL_COUNT unsigned int
: Number of basic blocks exited due to ifetch failure START_COMPILE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: Number of instructions compiled since START_COMPILE.

NONSEQ_COUNT unsigned int
: Number of basic blocks exited due to non-sequential instructions since START_COMPILE.

PAGE_STRADDLE_COUNT unsigned int
: Number of basic blocks exited due to unaligned instructions crossing page since START_COMPILE.

EXCEPTION

Exception taken. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

IS_PHYSICAL bool
: Physical or Virtual exception

LR unsigned int
: The value assigned to the link register.

PC unsigned int
: The location where the exception occurred.

PREFERRED_RETURN unsigned int
: The preferred return address for the exception.

TARGET_ISET enum
: The instruction set of the exception handler code.

TARGET_PC unsigned int
: The address the exception branches to.

VECTOR enum
: The exception vector.

VECTOR_OFFSET unsigned int
: The offset of an exception vector from the base address.

EXCEPTION_END

Every exception completed.

EXCEPTION_RAISE

Every exception raised.

EXCEPTION_RETURN

Branches on leaving exception. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

EXCEPTION_RETURN_PREBRANCH

Exception return event that must occur before PC and processor state have updated.

EXCEPTION_START

Every exception started.

FP_STATE

Floating point state. Fields:

FPSR unsigned int
: FPSR register value.

MASK unsigned int
: The mask indicating updated cumulative bits.

HLT

HLT instruction occurred. Fields:

IMM16 unsigned int
: The 16-bit immediate value encoded in HLT instruction.

INST

Every instruction executed. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DEBUG_STATE bool
: The instruction is executed in debug state.

DISASS string
: Disassembly of instruction.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The current instruction set.

ITSTATE unsigned int
: The current ITSTATE.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

MODE enum
: The mode the core is in.

NS unsigned int
: The core's non-secure bit.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

OPCODE unsigned int
: The opcode of the instruction.

PADDR unsigned int
: The physical address of the instruction.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the instruction.

PC unsigned int
: The address of the instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SIZE unsigned int
: The size of the instruction in bytes.

INST_END

Every instruction completed.

INST_START

Every instruction started. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

INST_STRADDLE

Instruction straddles boundary. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

ISB

ISB instruction occurred.

LOAD_MULTIPLE_REGS_LIST

List of destination registers (not source) of a multiple registers load instruction. Fields:

REG string
: Register to add into the list

LOCAL_MONITOR

Local monitor activity. Fields:

PADDR unsigned int
: Local Monitor Address

State enum
: State of the monitor (Open/Exclusive)

MEMMAP_DEBUG_READ

Memory mapped reads to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

MEMMAP_DEBUG_WRITE

Memory mapped writes to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

MMU_TRANS

Address translation information. Fields:

ASID unsigned int
: Address space identifier.

CORE_NUM unsigned int
: Core number in a multi processor.

Hyp bool
: Entry matches in Hyp state only

INNERCACHE_RA bool
: Is the inner cache allocate on read

INNERCACHE_TRANSIENT bool
: Is the inner write-through transient

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

MEMTYPE enum
: Memory type.

NSDESC enum
: Is secure side supposed to access secure or nonsecure memory

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TRANSIENT bool
: Is the outer write-through transient

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PADDR unsigned int
: Physical address of the access.

PAGESIZE unsigned int
: Page size as log2(size).

SH enum
: Shareability

SIDE enum
: Inst / Data.

STAGE1_PERM unsigned int
: Stage 1 permissions, [5:3] Privileged access (XWR) [2:0] User access (XWR)

STAGE2_PERM unsigned int
: Stage 2 permission mask [5:3] Privileged access (XWR) [2:0] User access (XWR)

VADDR unsigned int
: Virtual address of the access.

VMID unsigned int
: Virtual machine identifier.

XS bool
: XS Attribute

nG enum
: Flag indicating whether ASID will be matched

MMU_TTB_READ

This event is triggered by reads caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data read.

IPA unsigned int
: For stage 1, the IPA of the read.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

OUTADDR unsigned int
: The output physical address.

PADDR unsigned int
: The physical address of the read.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MMU_TTB_WRITE

This event is triggered by writes caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data written.

IPA unsigned int
: For stage 1, the IPA of the write.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

PADDR unsigned int
: The physical address of the write.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MODE_CHANGE

Mode change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: The new mode.

NON_SECURE enum
: The new security state

OLD_MODE enum
: The old mode.

NOT_TAG_CHECKED_INST

MTE Tag not checked for current instruction.

PERIODIC

Called for every quantum. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

PC unsigned int
: The address of the next instruction to be executed on this CPU.

PMU_COUNTER_OVERFLOW

PMU counter overflow. Fields:

EVENT_ID unsigned int
: Event Identifier

INDEX unsigned int
: Counter index (as selected by PMSELR).

INTERRUPT bool
: Is interrupt enabled on overflow for this counter

PRELOAD_DATA

Data preload from PLD instruction. Fields:

VADDR unsigned int
: Virtual address of the data that should be preloaded.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PRELOAD_INST

Instruction preload from PLI instruction. Fields:

VADDR unsigned int
: Virtual address of the instruction that should be preloaded.

PRE_CORE_LOAD

Trace just before a core load. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_CORE_STORE

Trace just before a core store. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_TTB_READ

This event is triggered before reads caused by a translation table walk. Fields:

ADDR unsigned int
: The physical address of the read.

PSTATE_PACTIVE_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

PSTATE_PACTIVE_OPMODE_TRANSITION_DENIED

Trace the denied pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_POWER_TRANSITION

Trace pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OLD_STATE enum
: Power mode before the transition

PSTATE_PACTIVE_POWER_TRANSITION_DENIED

Trace the denied pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_THREAD_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions for threads on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

RUN_STATE

Run state transition. Fields:

INST_COUNT unsigned int
: Ticks count at point of transition.

NEW enum
: New run state.

OLD enum
: Old run state.

SEMIHOSTING_CALL

Call of a semihost function occurred. Fields:

PC unsigned int
: The program counter after the semihosting call.

SEMIHOSTING_PRECALL

About to call semihost. Fields:

EL2 bool
: Use EL2 translation regime.

NS bool
: Is Non-Secure.

REG_WIDTH unsigned int
: The current register width in bytes.

SIGNAL

External signal state change. Fields:

SIGNAL enum
: Signal that changed

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

STATE bool
: Signal asserted state

SPSR

SPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: Which of the banked SPSR registers is written.

OLD_VALUE unsigned int
: The old SPSR value

VALUE unsigned int
: The new SPSR value

START_COMPILE

Compilation started. Fields:

VADDR unsigned int
: Instruction where compilation begins.

STG_MULTIPLE_MICRO_OPS_ADDRESS_RANGE

Address range for MTE stores with multiple uops. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

STORE_MULTIPLE_REGS_ADDRESS_RANGE

Address range for stores with multiple registers. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

SUBOPTIMAL_LDST_RETIRED

suboptimal load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1: half-word, 2: word ...)

OPTIMAL_ALIGN_SELECTOR unsigned int
: Selector to decide the optimal alignment size (1: access width, 2: 4-bytes, 3: 8-bytes, ..., 12: 4-kbytes)

VADDR unsigned int
: The virtual address of the access

SYNC

Called for every synchronization, that is every quantum break. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

LOCAL_QUANTUM unsigned int
: The local quantum of this CPU.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOCAL_TIME unsigned int
: The local time of this CPU.

SYNC_REASON enum
: Reason for sync.

SYSCALL

System call instruction executed. Fields:

IMM unsigned int
: Immediate value of the system call instruction.

TYPE enum
: System call type.

VADDR unsigned int
: Instruction that caused the system call.

SYSREG_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE32

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE64

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

UNALIGNED_LDST_RETIRED

Processor unaligned load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1=>half-word, 2->word ...)

NUMBER_OF_BEATS unsigned int
: Number of accesses (beats) in this burst (Total size in bytes is ACCESS_SIZE * NUMBER_OF_BEATS)

VADDR unsigned int
: The virtual address of the access

VFP_D_REGS

VFP/NEON D 64 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_Q_REGS

VFP/NEON Q 128 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding byte in the register. For example, 0x1 would indicate a write to VALUE[7:0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_SYS_REGS

Writes to the VFP/NEON units system registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: Which VFP system register is written.

MASK unsigned int
: The mask indicating updated cumulative bits.

OLD_VALUE unsigned int
: The register's old value overwritten.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VALUE unsigned int
: The new value written to the VFP system register.

VFP_S_REGS

VFP/NEON S 32 bit register write. Fields:

ALIASING enum
: Type of register aliasing used.

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

WAYPOINT

Signals end of basic block execution. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: Origin instructions set

IS_COND bool
: Indicates if this is a conditional waypoint.

PC unsigned int
: Origin address (or 0 if unavailable)

TAKEN bool
: Indicates if this waypoint was taken.

TARGET unsigned int
: Destination address

TARGET_ISET enum
: Destination instructions set

WFE_END

WFE ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFE.

WFE_EVENT_REGISTER

WFE event register status: set/clear, reason. Fields:

INST_COUNT unsigned int
: Ticks count.

REASON enum
: Reason for set/clear. Only REASON==0 (Cleared by WFE) clears the bit, all other reasons set it.

WFE_IGNORED

WFE ignored. Fields:

EVENT bool
: This WFE was ignored because the event register was set.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: Ticks count when ignoring WFE.

TRAPPED bool
: This WFE was trapped.

WFE_START

WFE entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFE.

WFI_END

WFI ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFI.

WFI_IGNORED

WFI ignored. Fields:

DISABLED bool
: This WFI was ignored because WFI is disabled.

INST_COUNT unsigned int
: Ticks count when ignoring WFI.

REASON enum
: specifies reason why WFI trace was ignored

TRAPPED bool
: This WFI was trapped.

WFI_START

WFI entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFI.

WFI_WAKEUP

WFI wakeup. Fields:

INST_COUNT unsigned int
: Ticks count when WFI wakeup occurred.

REASON enum
: Reason for wakeup.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.7 ARM_Cortex-A17 - trace

This section describes the trace sources.

ASYNC_MEMORY_FAULT

Context ID Register write. Fields:

FAULT unsigned int
: Fault status in ESR format

PADDR unsigned int
: Physical Address (or 0 if unavailable)

VADDR signed int
: Virtual Address (or 0 if unavailable)

ATOMIC_END_ACCESS

Bus trace access for atomics. Fields:

ACCESS_FAIL bool
: Memory access failed

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

LOAD_VALUE unsigned int
: Loaded values

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ATOMIC_START_ACCESS

Bus trace access for atomics. Fields:

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

NSDESC unsigned int
: The security state of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ArchMsg.Error.exit_code

Fields:

EXITCODE signed int
: Exit code to be returned

KIND string
: Component kind that invoked the exit code trace

REASON string
: Reason why the exit code trace was invoked (optional)

ArchMsg.Warning.IgnoredWarmResetRequest

Event fires when software attempts to request warm reset through write to deprecated DBGPRCR.CWRR.

ArchMsg.Warning.TarmacReconstructor

Unexpected event happened during Tarmac reconstruction. Fields:

REASON string
: Reason

ArchMsg.Warning.Unpredictably Indexed PM Event Register

Fields:

IsDirect unsigned int
: Direct Access rather than indirect

N unsigned int
: Number of Accessible Registers

SEL unsigned int
: Selector

ArchMsg.Warning.branch_to_unaligned_address

Fields:

ADDR unsigned int
: Unaligned branch address

ArchMsg.Warning.cache_contents_unknown

A cache has been activated without being invalidated and may contain UNKNOWN content. Fields:

REGIME unsigned int
: Current translation regime

SIDE unsigned int
: 0 d-side cache or unified cache, 1 i-side

ArchMsg.Warning.dcimvac_matches_watchpoint

Watchpoints matching an AArch32 DCIMVA are implementation defined.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_fieldmismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_fracbitsnegative

VCVT Instruction is unpredictable with a negative number of fraction bits.

ArchMsg.Warning.decode_initblock

Instruction is unpredictable in an IT block.

ArchMsg.Warning.decode_initblocknotlast

Instruction is unpredictable in an IT block when not the last.

ArchMsg.Warning.decode_invalid_condition

Instruction is unpredictable when the condition is not AL.

ArchMsg.Warning.decode_invalidfieldcombination

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_invalidvalue

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_mem_hint_unallocated

There is no memory hint allocated to this bit pattern.

ArchMsg.Warning.decode_registermatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registermismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registeroutofrange

Float point instruction is unpredictable indexing beyond the end of the register bank.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_sbzsbo

Reserved bits in the instruction are not canonical.

ArchMsg.Warning.decode_transactiontoobig

Attempt to perform a memory transaction of more than 128 bytes.

ArchMsg.Warning.decode_unpred_other

Instruction is unpredictable.

ArchMsg.Warning.decode_unpreduseofpc

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13inreglist

Instruction is unpredictable with R13 in register-list.

ArchMsg.Warning.decode_useofd16orhigher

Attempt to use register D16 or higher where it is not allowed.

ArchMsg.Warning.decode_writebackandbaseinlist

Instruction is unpredictable with write back when the base register is in the transfer list.

ArchMsg.Warning.decode_zeroregistersinlist

Instruction is unpredictable when no registers are to be transfered.

ArchMsg.Warning.recursive_branch

An instruction is performing a branch that targets the same instruction. If this was intended then a WFI might be more effective. Fields:

PC unsigned int
: Address of instruction causing the branch

ArchMsg.Warning.recursive_exception

An instruction has generated an exception that targets the same instruction. Fields:

CPSR unsigned int
: Processor state

PC unsigned int
: Address of instruction causing the exception

TYPE enum
: Type of exception

ArchMsg.Warning.reentrant_vector_catch

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR unsigned int
vector address

ArchMsg.Warning.reserved_it_state

Fields:

ITSTATE unsigned int
reserved IT state

ArchMsg.Warning.tlb_contents_unknown

Translation has been enabled without invalidating the TLB. Fields:

EL unsigned int
Exception level of translation regime
INVALIDITY unsigned int
Which TLB may be invalid (if TLBs are separate)
NON_SECURE unsigned int
Security state of translation regime
VMID unsigned int
Current VMID, if applicable

ArchMsg.Warning.unpredictable_a32_breakpoint

Fields:

ADDR1 unsigned int
Address
ADDR2 unsigned int
Breakpoint address
BAS unsigned int
BAS Field
IS_ADDRESS_MISMATCH_BREAKPOINT bool
Is Mismatch breakpoint, not a match breakpoint
N unsigned int
Breakpoint number
OFFSET unsigned int
Indicates which halfword actually considering into the A32 instruction

ArchMsg.Warning.unpredictable_banked_register_access

Fields:

R unsigned int
R bit
SYSM unsigned int
SYSm

ArchMsg.Warning.unpredictable_ccfail_undef

Fields:

OPCODE unsigned int
opcode of the instruction
VADDR unsigned int
Virtual address of the instruction being executed

ArchMsg.Warning.unpredictable_ici_bits

ICI bits do not apply or are architecturally unpredictable for instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.unpredictable_t32_breakpoint

Fields:

ADDR1 unsigned int
: Address

ADDR2 unsigned int
: Breakpoint address

BAS unsigned int
: BAS Field

IS_ADDRESS_MISMATCH_BREAKPOINT bool
: Is Mismatch breakpoint, not a match breakpoint

N unsigned int
: Breakpoint number

OFFSET unsigned int
: Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_t32_vector_catch

Fields:

ADDR unsigned int
: Address

OFFSET unsigned int
: Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_watchpoint_far

Fields:

LOWER_BOUND unsigned int
: the lowest address accessed by the instruction that triggered the watchpoint

UPPER_BOUND unsigned int
: the highest watchpointed address accessed by that instruction

ArchMsg.Warning.warning_AdvSIMDExpandImmUnexpectedZero

AdvSIMDExpandImm may treat this immediate value as UNPREDICTABLE.

ArchMsg.Warning.warning_ConditionalSMC

SMC instruction has UNPREDICTABLE effects when conditional when combined with traps.

ArchMsg.Warning.warning_access_crosses_page_boundary

Access crosses page boundary.

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault_on_both_pages

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_spanning_different_memory

Fields:

ADDR unsigned int
: address of access crossing page

MEMTYPE_PAGE1 signed int
: First Page's memory type (0-Normal, 1-Non-Normal)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MEMTYPE_PAGE2 signed int
Second Page's memory type (0-Normal, 1-Non-Normal)

ArchMsg.Warning.warning_access_wraps_around_memory

Fields:

WIDTH unsigned int
Address width

ArchMsg.Warning.warning_bx_from_thumbee

Fields:

ADDR unsigned int
Destination address

ArchMsg.Warning.warning_ccsidr_unimplemented_level

Fields:

CSSELR unsigned int
current effective value of CSSELR

ArchMsg.Warning.warning_change_to_ns_when_tge_set

Attempting to change NS to 1 when HCR.TGE = 1.

ArchMsg.Warning.warning_contiguous_bit_check_abort

Fields:

ENTRY_ADDR unsigned int
address of conflicting TLB entry

ArchMsg.Warning.warning_contiguous_bit_error

Fields:

CONTIG_ADDR unsigned int
address of the TLB first read, with which this entry is expected to be contiguous
CONTIG_DATA unsigned int
expected contents based on the entry at CONTIG_ADDR
ENTRY_ADDR unsigned int
address of conflicting TLB entry
ENTRY_DATA unsigned int
contents of conflicting TLB entry

ArchMsg.Warning.warning_cp10_cp11_mismatch

Fields:

EL unsigned int
EL owning the control

ArchMsg.Warning.warning_cp10_cp11_reserved_value

Fields:

CP10 unsigned int
CP10 access
CP11 unsigned int
CP11 access

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_csselr_level_out_of_range

Fields:

IMPLEMENTED unsigned int
: number of cache levels implemented

LEVEL unsigned int
: written value of CSSELR.Level

TYPE enum
: The type of cache selected by CSSELR.InD and TnD

ArchMsg.Warning.warning_debug_flow_control_bits_not_obeyed

Fields:

ERROR enum
: type of error resulting

REGISTER enum
: register under consideration

ArchMsg.Warning.warning_debug_register_access_during_reset

Fields:

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

ArchMsg.Warning.warning_decode_cps_inconsistent_fields

The mode setting fields in the CPS instruction are inconsistent.

ArchMsg.Warning.warning_decode_invalid_state

Instruction is unpredictable in current exception-level/security state.

ArchMsg.Warning.warning_default_cacheable_mmu_on

Default cacheable is enabled (HCR.DC = 1) while MMU is enabled.

ArchMsg.Warning.warning_default_cacheable_vmmu_off

Default cacheable is enabled (HCR.DC = 1) while HCR.VM = 0.

ArchMsg.Warning.warning_deprecated_wvr_bit_2_set

Fields:

N unsigned int
: index

WVR unsigned int
: DBGWVR

ArchMsg.Warning.warning_eret_while_software_step_active_pending

Missing ISB between setting MDSCR_EL1.SS and ERET.

ArchMsg.Warning.warning_exclusive_to_non_normal

Fields:

ADDR unsigned int
: Address

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_exclusive_to_non_writeback

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_execute_from_device_memory

An attempt was made to execute from Device memory.

ArchMsg.Warning.warning_illegal_cpsr_mode

Fields:

NEW_MODE unsigned int
: New value of CPSR.M

ArchMsg.Warning.warning_illegal_srs_mode

Fields:

MODE unsigned int
: mode for Banked SP

ArchMsg.Warning.warning_implementation_defined_read_debug_register_in_SCS

Read Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_implementation_defined_sequential_security_transitions_supported

The behaviour of a sequential instruction fetches that cross from non-secure to secure memory and contain SG instruction is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_implementation_defined_stack_limit_check_supported

It is IMPLEMENTATION DEFINED whether stack pointer limit checking is performed for the this instructions. Fields:

ADDRESS unsigned int
: Address of instruction

OPCODE unsigned int
: instruction opcode

ArchMsg.Warning.warning_implementation_defined_write_debug_register_in_SCS

Write Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

DATA unsigned int
: data attempted to be written

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_invalid_tcr_granule

Fields:

REQUEST signed int
: page size requested (or 0 for reserved)

SUBSTITUTE signed int
: best guess available page size

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

TG_ID bool
: bits TG0 or TG1

ArchMsg.Warning.warning_load_multiple_user_registers_from_user_mode

An LDM(user registers) instruction executed from user mode.

ArchMsg.Warning.warning_load_pc_from_unaligned

PC loaded from an unaligned location.

ArchMsg.Warning.warning_mte_in_mem_attr_not_supported

Fields:

ATTR unsigned int
: Memory attribute

ArchMsg.Warning.warning_reserved_breakpoint_state_match

Fields:

IS_BREAKPOINT unsigned int
: Is breakpoint, not watchpoint

ArchMsg.Warning.warning_shareability

Unpredictable: combination of the 1st and 2nd stages of translation is Normal Inner Non-Cacheable, Outer Non-Cacheable.

ArchMsg.Warning.warning_software_step_set_while_enabled

MDSCR_EL1.SS set to 1 while software step debug exceptions are enabled.

ArchMsg.Warning.warning_thumb_instruction_wraps_around_memory

Thumb instruction warps around memory.

ArchMsg.Warning.warning_ttbr_sbz_bits_are_not_zero

Fields:

MSB unsigned int
: x-1

TTBR enum
: which TTBR register

ArchMsg.Warning.warning_unaligned_address_dbgdtrrx_write

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_unaligned_address_dbgdtrtx_read

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_unaligned_to_device

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_strongly_ordered

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unknown_sau_rnr

SAU_RNR was set to an unsupported value.

ArchMsg.Warning.warning_unpredictable_AIRCR_PRIP_and_BFHFNMINP

The effect of setting both AIRCR.BFHFNMINP and AIRCR.PRIP to 1 is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTCLRACTIVE_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTCLRACTIVE if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_and_SYSRESETREQ

When the processor is halted in Debug state, if a write to the register writes a 1 to both VECTRESET and SYSRESETREQ, the behavior is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTRESET if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_incorrect_VKEY

The value 0x05FA must be written to AIRCR.VKEY, otherwise the register write is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_EXE_RETURN_Reserved_Bit

EXC_RETURN[23:7] are reserved with the special condition that all bits should be written as one. Values other than all 1s are UNPREDICTABLE. Fields:

EXC_RETURN unsigned int
Special PC value
SBO_MASK unsigned int
Which bits Should Be One to not UNPRED

ArchMsg.Warning.warning_unpredictable_change_to_DEMCR_MON_STEP_at_insufficient_priority

The effect of changing DEMCR.MON_STEP at an execution priority that is lower than the priority of the DebugMonitor exception is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_change_to_priority_of_active_exception

Changing the priority of an active exception is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_clear_lspact

Fields:

DRVEXC_TAKEN unsigned int
derived exception taken not pended

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

REACHED_LAST_STORE unsigned int
only the last faulted

ArchMsg.Warning.warning_unpredictable_dbg_exit_unaligned_dlr

Debug exit to AArch32 with DLR[0] set to 1. PC[0] can be set to 0 or to DLR[0].

ArchMsg.Warning.warning_unpredictable_exception_catch

Generation of an exception catch event is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_inconsistent_state

state on exception return is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_instruction

exception return instruction is unpredictable.

ArchMsg.Warning.warning_unpredictable_in_debug_state

Instruction is UNPREDICTABLE when executed in debug state.

ArchMsg.Warning.warning_unpredictable_mair_encoding

Fields:

ATTR unsigned int
Memory attribute

ArchMsg.Warning.warning_unpredictable_pmu_counter_access

Access to PMU counters from non-secure EL0 or EL1 is UNPREDICTABLE with MDCR_EL2.HPMN set to 0.

ArchMsg.Warning.warning_unpredictable_prioritization_breakpoint_match_vector_catch

Two events with the same priority occurred at the same instruction: Address Matching Vector Catch debug event, and Breakpoint debug event It is constrained unpredictable which is taken.

ArchMsg.Warning.warning_unpredictable_stack_selection

UNPREDICTABLE state on context switch.

ArchMsg.Warning.warning_unpredictable_tsize_out_of_range

Fields:

TCR enum
Which TCR
TSIZE signed int
Value of TSize
TSIZE_MAX signed int
The maximum allowed value
TTBR unsigned int
Which TTBR region

ArchMsg.Warning.warning_unpredictable_unaligned_pc_as_base_register

The PC value must be word-aligned, otherwise the behavior of the instruction is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_unaligned_pop_stack

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

FRAME_ADDR unsigned int
: unaligned stack frame address

ArchMsg.Warning.warning_unpredictable_vmsa_memattrib

Fields:

TEXCB unsigned int
: texcb value

TEX_REMAP bool
: use tex remap

ArchMsg.Warning.warning_unpredictable_vtcr_t0sz_sl0

Fields:

SL0 signed int
: VTCR.SL0, starting level for stage 2 translation table walks

T0SZ signed int
: VTCR.T0SZ, required input address range

ArchMsg.Warning.warning_unpredictable_write_DHCSR

Fields:

NEW unsigned int
: data attempted to be written

OLD unsigned int
: previous value

ArchMsg.Warning.warning_unpredictable_write_SHCSR

UNPREDICTABLE setting a pending bit to 1 for an exception with priority >= execution priority.

ArchMsg.Warning.warning_unsupported_access_to_memory_mapped_register

Access to memory mapped register is unsupported.

ArchMsg.Warning.warning_user_jmcr_access

Fields:

WRITE unsigned int
: access is write

ArchMsg.Warning.warning_wcr_mask_and_bas

DBGWCR_EL1.MASK is non-zero and BAS is not 0xff.

ArchMsg.Warning.warning_wcr_mask_reserved

DBGWCR_ELn.MASK is set to a reserved value.

ArchMsg.Warning.warning_wcr_non_configuous_bas

DBGWCR_EL1.BAS has non-contiguous set of ones.

ArchMsg.Warning.warning_wvr_masked_bit_not_zero

DBGWVRn_EL1.VA contains a bit masked by DBGWCRn_EL1.MASK that is not zero.

BRANCH_MISPREDICT

Simulating branch mispredict. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PC unsigned int
: Origin address (or 0 if unavailable)

BRA_DIR

Direct branches, to immediate address. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR

Indirect branches, perhaps to a register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR_CCFAIL

Indirect branches fail, perhaps to a register.

BRA_RET_CCFAIL

Branche return fail, perhaps to a register.

CACHE_MAINTENANCE_OP

Cache Maintenance Operation. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Specified MVA or Set/way

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

FUNCTION enum
: Clean / Invalidate

SCOPE enum
: Affected region

SIDE enum
: Inst / Data.

CCFAIL

Conditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCFAIL_UNC

Unconditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS

Conditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS_UNC

Unconditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CODE_CACHE_MAINT

Code cache maintenance. Fields:

END_INCL unsigned int
: Inclusive end addres for by VA or PA requests

NS bool
: Non-secure world for PA

START unsigned int
: Start address for by VA or PA requests

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

TYPE enum
: Request type.

COMPILE_BLOCK_END

Last instruction of basic block translated. Fields:

VADDR unsigned int
: Address of next instruction after this basic block

COMPILE_BLOCK_START

First instruction of basic block translated. Fields:

VADDR unsigned int
: Address of first instruction in basic block

COMPILE_INST

ARM instruction compiled. Fields:

DISASS string
: Disassembly of the instruction.

ISET enum
: The instruction set of this instruction.

ITSTATE unsigned int
: The ITSTATE current for the instruction.

OPCODE unsigned int
: The opcode of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

CONTEXTIDR

Context ID Register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

NS bool
: Secure or nonsecure banked register is accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

CONTEXT_SYNC

Called for every context synchronization event. Fields:

ARCH_CSE bool
: Architectural Context Synchronization Event

SYSREG_SYNC bool
: System register synchronization event

CORE_ENDIAN

Core BE8 Big-Endian state changed. Fields:

BE8 bool
: Core BE8 Big-Endian state

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_INFO

Static processor attributes. Only triggered by a call to DumpState(). Fields:

ARCH_PROFILE enum
: The architecture profile of the core.

CLUSTER_ID unsigned int
: The cluster ID of this processor.

CORE_NUM unsigned int
: The number of this core in an MP processor.

DETAILED_CORE_FEATURES enum
: Detailed core features as a bitfield. Enum values are bitmasks to extract the info from the field

FPU_VERSION enum
: The VFP version implemented by the core.

MEM_ARCH enum
: The memory architecture of the core.

MIDR_EL1 unsigned int
: MIDR_EL1 value

NUM_CORES unsigned int
: The number of cores in this MP processor.

QUANTUM_SIZE unsigned int
: The default quantum size of the core.

SECURITY_FEATURES bool
: Does the core have security features?

CORE_LOADS

Processor load accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CORE_REGS

Changes of the core registers R0 to R14. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number, 0 to 14.

MODE enum
: Bank of the register accessed.

OLD_VALUE unsigned int
: The old value overwritten.

PHYS_ID enum
: The physical register accessed.

VALUE unsigned int
: The new value written to the register.

CORE_STORES

Processor store accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CP14_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP14_WRITE

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

CP15_WRITE

System Control Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

CPSR

CPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

OLD_VALUE unsigned int
: The old CPSR value

UNKNOWN unsigned int
: Bits within the register that have unknown value.

VALUE unsigned int
: The new CPSR value

CRYPTO_SPEC

Every crypto instruction speculatively executed.

CorePowerStateChange

Core power-up state has changed. Fields:

PU bool
: Whether the core powered up

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DATA_CACHE_SET_ALLOC_TAGS

Set allocation tags from DC GVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of memory in bytes.

TAG unsigned int
: The value of the tag being set.

VADDR unsigned int
: Virtual address used for setting allocation tags.

DATA_CACHE_ZERO

Zero and invalidate cache line from DC ZVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of zero'd memory in bytes.

VADDR unsigned int
: Virtual address of the zero'd cache line.

DEBUG_EVENT

Hardware debug support event. Fields:

EVENT enum
: Description of event

VALUE unsigned int
: data value

END_COMPILE

Compilation end. Fields:

END_OF_PAGE_COUNT unsigned int
: Number of basic blocks exited due to page boundary since START_COMPILE.

FETCHFAIL_COUNT unsigned int
: Number of basic blocks exited due to ifetch failure START_COMPILE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: Number of instructions compiled since START_COMPILE.

NONSEQ_COUNT unsigned int
: Number of basic blocks exited due to non-sequential instructions since START_COMPILE.

PAGE_STRADDLE_COUNT unsigned int
: Number of basic blocks exited due to unaligned instructions crossing page since START_COMPILE.

EXCEPTION

Exception taken. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

IS_PHYSICAL bool
: Physical or Virtual exception

LR unsigned int
: The value assigned to the link register.

PC unsigned int
: The location where the exception occurred.

PREFERRED_RETURN unsigned int
: The preferred return address for the exception.

TARGET_ISET enum
: The instruction set of the exception handler code.

TARGET_PC unsigned int
: The address the exception branches to.

VECTOR enum
: The exception vector.

VECTOR_OFFSET unsigned int
: The offset of an exception vector from the base address.

EXCEPTION_END

Every exception completed.

EXCEPTION_RAISE

Every exception raised.

EXCEPTION_RETURN

Branches on leaving exception. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

EXCEPTION_RETURN_PREBRANCH

Exception return event that must occur before PC and processor state have updated.

EXCEPTION_START

Every exception started.

FP_STATE

Floating point state. Fields:

FPSR unsigned int
: FPSR register value.

MASK unsigned int
: The mask indicating updated cumulative bits.

HLT

HLT instruction occurred. Fields:

IMM16 unsigned int
: The 16-bit immediate value encoded in HLT instruction.

INST

Every instruction executed. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DEBUG_STATE bool
: The instruction is executed in debug state.

DISASS string
: Disassembly of instruction.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The current instruction set.

ITSTATE unsigned int
: The current ITSTATE.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

MODE enum
: The mode the core is in.

NS unsigned int
: The core's non-secure bit.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

OPCODE unsigned int
: The opcode of the instruction.

PADDR unsigned int
: The physical address of the instruction.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the instruction.

PC unsigned int
: The address of the instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SIZE unsigned int
: The size of the instruction in bytes.

INST_END

Every instruction completed.

INST_START

Every instruction started. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

INST_STRADDLE

Instruction straddles boundary. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

ISB

ISB instruction occurred.

LOAD_MULTIPLE_REGS_LIST

List of destination registers (not source) of a multiple registers load instruction. Fields:

REG string
: Register to add into the list

LOCAL_MONITOR

Local monitor activity. Fields:

PADDR unsigned int
: Local Monitor Address

State enum
: State of the monitor (Open/Exclusive)

MEMMAP_DEBUG_READ

Memory mapped reads to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

MEMMAP_DEBUG_WRITE

Memory mapped writes to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

MMU_TRANS

Address translation information. Fields:

ASID unsigned int
: Address space identifier.

CORE_NUM unsigned int
: Core number in a multi processor.

Hyp bool
: Entry matches in Hyp state only

INNERCACHE_RA bool
: Is the inner cache allocate on read

INNERCACHE_TRANSIENT bool
: Is the inner write-through transient

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

MEMTYPE enum
: Memory type.

NSDESC enum
: Is secure side supposed to access secure or nonsecure memory

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TRANSIENT bool
: Is the outer write-through transient

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PADDR unsigned int
: Physical address of the access.

PAGESIZE unsigned int
: Page size as log2(size).

SH enum
: Shareability

SIDE enum
: Inst / Data.

STAGE1_PERM unsigned int
: Stage 1 permissions, [5:3] Privileged access (XWR) [2:0] User access (XWR)

STAGE2_PERM unsigned int
: Stage 2 permission mask [5:3] Privileged access (XWR) [2:0] User access (XWR)

VADDR unsigned int
: Virtual address of the access.

VMID unsigned int
: Virtual machine identifier.

XS bool
: XS Attribute

nG enum
: Flag indicating whether ASID will be matched

MMU_TTB_READ

This event is triggered by reads caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data read.

IPA unsigned int
: For stage 1, the IPA of the read.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

OUTADDR unsigned int
: The output physical address.

PADDR unsigned int
: The physical address of the read.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MMU_TTB_WRITE

This event is triggered by writes caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data written.

IPA unsigned int
: For stage 1, the IPA of the write.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

PADDR unsigned int
: The physical address of the write.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MODE_CHANGE

Mode change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: The new mode.

NON_SECURE enum
: The new security state

OLD_MODE enum
: The old mode.

NOT_TAG_CHECKED_INST

MTE Tag not checked for current instruction.

PERIODIC

Called for every quantum. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

PC unsigned int
: The address of the next instruction to be executed on this CPU.

PMU_COUNTER_OVERFLOW

PMU counter overflow. Fields:

EVENT_ID unsigned int
: Event Identifier

INDEX unsigned int
: Counter index (as selected by PMSELR).

INTERRUPT bool
: Is interrupt enabled on overflow for this counter

PRELOAD_DATA

Data preload from PLD instruction. Fields:

VADDR unsigned int
: Virtual address of the data that should be preloaded.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PRELOAD_INST

Instruction preload from PLI instruction. Fields:

VADDR unsigned int
: Virtual address of the instruction that should be preloaded.

PRE_CORE_LOAD

Trace just before a core load. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_CORE_STORE

Trace just before a core store. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_TTB_READ

This event is triggered before reads caused by a translation table walk. Fields:

ADDR unsigned int
: The physical address of the read.

PSTATE_PACTIVE_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

PSTATE_PACTIVE_OPMODE_TRANSITION_DENIED

Trace the denied pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_POWER_TRANSITION

Trace pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OLD_STATE enum
: Power mode before the transition

PSTATE_PACTIVE_POWER_TRANSITION_DENIED

Trace the denied pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_THREAD_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions for threads on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

RUN_STATE

Run state transition. Fields:

INST_COUNT unsigned int
: Ticks count at point of transition.

NEW enum
: New run state.

OLD enum
: Old run state.

SEMIHOSTING_CALL

Call of a semihost function occurred. Fields:

PC unsigned int
: The program counter after the semihosting call.

SEMIHOSTING_PRECALL

About to call semihost. Fields:

EL2 bool
: Use EL2 translation regime.

NS bool
: Is Non-Secure.

REG_WIDTH unsigned int
: The current register width in bytes.

SIGNAL

External signal state change. Fields:

SIGNAL enum
: Signal that changed

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

STATE bool
: Signal asserted state

SPSR

SPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: Which of the banked SPSR registers is written.

OLD_VALUE unsigned int
: The old SPSR value

VALUE unsigned int
: The new SPSR value

START_COMPILE

Compilation started. Fields:

VADDR unsigned int
: Instruction where compilation begins.

STG_MULTIPLE_MICRO_OPS_ADDRESS_RANGE

Address range for MTE stores with multiple uops. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

STORE_MULTIPLE_REGS_ADDRESS_RANGE

Address range for stores with multiple registers. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

SUBOPTIMAL_LDST_RETIRED

suboptimal load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1: half-word, 2: word ...)

OPTIMAL_ALIGN_SELECTOR unsigned int
: Selector to decide the optimal alignment size (1: access width, 2: 4-bytes, 3: 8-bytes, ..., 12: 4-kbytes)

VADDR unsigned int
: The virtual address of the access

SYNC

Called for every synchronization, that is every quantum break. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

LOCAL_QUANTUM unsigned int
: The local quantum of this CPU.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOCAL_TIME unsigned int
: The local time of this CPU.

SYNC_REASON enum
: Reason for sync.

SYSCALL

System call instruction executed. Fields:

IMM unsigned int
: Immediate value of the system call instruction.

TYPE enum
: System call type.

VADDR unsigned int
: Instruction that caused the system call.

SYSREG_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE32

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE64

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

UNALIGNED_LDST_RETIRED

Processor unaligned load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1=>half-word, 2->word ...)

NUMBER_OF_BEATS unsigned int
: Number of accesses (beats) in this burst (Total size in bytes is ACCESS_SIZE * NUMBER_OF_BEATS)

VADDR unsigned int
: The virtual address of the access

VFP_D_REGS

VFP/NEON D 64 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_Q_REGS

VFP/NEON Q 128 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding byte in the register. For example, 0x1 would indicate a write to VALUE[7:0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_SYS_REGS

Writes to the VFP/NEON units system registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: Which VFP system register is written.

MASK unsigned int
: The mask indicating updated cumulative bits.

OLD_VALUE unsigned int
: The register's old value overwritten.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VALUE unsigned int
: The new value written to the VFP system register.

VFP_S_REGS

VFP/NEON S 32 bit register write. Fields:

ALIASING enum
: Type of register aliasing used.

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

WAYPOINT

Signals end of basic block execution. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: Origin instructions set

IS_COND bool
: Indicates if this is a conditional waypoint.

PC unsigned int
: Origin address (or 0 if unavailable)

TAKEN bool
: Indicates if this waypoint was taken.

TARGET unsigned int
: Destination address

TARGET_ISET enum
: Destination instructions set

WFE_END

WFE ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFE.

WFE_EVENT_REGISTER

WFE event register status: set/clear, reason. Fields:

INST_COUNT unsigned int
: Ticks count.

REASON enum
: Reason for set/clear. Only REASON==0 (Cleared by WFE) clears the bit, all other reasons set it.

WFE_IGNORED

WFE ignored. Fields:

EVENT bool
: This WFE was ignored because the event register was set.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: Ticks count when ignoring WFE.

TRAPPED bool
: This WFE was trapped.

WFE_START

WFE entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFE.

WFI_END

WFI ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFI.

WFI_IGNORED

WFI ignored. Fields:

DISABLED bool
: This WFI was ignored because WFI is disabled.

INST_COUNT unsigned int
: Ticks count when ignoring WFI.

REASON enum
: specifies reason why WFI trace was ignored

TRAPPED bool
: This WFI was trapped.

WFI_START

WFI entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFI.

WFI_WAKEUP

WFI wakeup. Fields:

INST_COUNT unsigned int
: Ticks count when WFI wakeup occurred.

REASON enum
: Reason for wakeup.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.8 ARM_Cortex-A32 - trace

This section describes the trace sources.

ASYNC_MEMORY_FAULT

Context ID Register write. Fields:

FAULT unsigned int
: Fault status in ESR format

PADDR unsigned int
: Physical Address (or 0 if unavailable)

VADDR signed int
: Virtual Address (or 0 if unavailable)

ATOMIC_END_ACCESS

Bus trace access for atomics. Fields:

ACCESS_FAIL bool
: Memory access failed

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

LOAD_VALUE unsigned int
: Loaded values

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ATOMIC_START_ACCESS

Bus trace access for atomics. Fields:

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

NSDESC unsigned int
: The security state of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ArchMsg.Error.exit_code

Fields:

EXITCODE signed int
: Exit code to be returned

KIND string
: Component kind that invoked the exit code trace

REASON string
: Reason why the exit code trace was invoked (optional)

ArchMsg.Warning.IgnoredWarmResetRequest

Event fires when software attempts to request warm reset through write to deprecated DBGPRCR.CWRR.

ArchMsg.Warning.TarmacReconstructor

Unexpected event happened during Tarmac reconstruction. Fields:

REASON string
: Reason

ArchMsg.Warning.Unpredictably Indexed PM Event Register

Fields:

IsDirect unsigned int
: Direct Access rather than indirect

N unsigned int
: Number of Accessible Registers

SEL unsigned int
: Selector

ArchMsg.Warning.branch_to_unaligned_address

Fields:

ADDR unsigned int
: Unaligned branch address

ArchMsg.Warning.cache_contents_unknown

A cache has been activated without being invalidated and may contain UNKNOWN content. Fields:

REGIME unsigned int
: Current translation regime

SIDE unsigned int
: 0 d-side cache or unified cache, 1 i-side

ArchMsg.Warning.dcimvac_matches_watchpoint

Watchpoints matching an AArch32 DCIMVA are implementation defined.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_fieldmismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_fracbitsnegative

VCVT Instruction is unpredictable with a negative number of fraction bits.

ArchMsg.Warning.decode_initblock

Instruction is unpredictable in an IT block.

ArchMsg.Warning.decode_initblocknotlast

Instruction is unpredictable in an IT block when not the last.

ArchMsg.Warning.decode_invalid_condition

Instruction is unpredictable when the condition is not AL.

ArchMsg.Warning.decode_invalidfieldcombination

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_invalidvalue

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_mem_hint_unallocated

There is no memory hint allocated to this bit pattern.

ArchMsg.Warning.decode_registermatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registermismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registeroutofrange

Float point instruction is unpredictable indexing beyond the end of the register bank.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_sbzsbo

Reserved bits in the instruction are not canonical.

ArchMsg.Warning.decode_transactiontoobig

Attempt to perform a memory transaction of more than 128 bytes.

ArchMsg.Warning.decode_unpred_other

Instruction is unpredictable.

ArchMsg.Warning.decode_unpreduseofpc

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13inreglist

Instruction is unpredictable with R13 in register-list.

ArchMsg.Warning.decode_useofd16orhigher

Attempt to use register D16 or higher where it is not allowed.

ArchMsg.Warning.decode_writebackandbaseinlist

Instruction is unpredictable with write back when the base register is in the transfer list.

ArchMsg.Warning.decode_zeroregistersinlist

Instruction is unpredictable when no registers are to be transfered.

ArchMsg.Warning.recursive_branch

An instruction is performing a branch that targets the same instruction. If this was intended then a WFI might be more effective. Fields:

PC unsigned int
: Address of instruction causing the branch

ArchMsg.Warning.recursive_exception

An instruction has generated an exception that targets the same instruction. Fields:

CPSR unsigned int
: Processor state

PC unsigned int
: Address of instruction causing the exception

TYPE enum
: Type of exception

ArchMsg.Warning.reentrant_vector_catch

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR unsigned int
vector address

ArchMsg.Warning.reserved_it_state

Fields:

ITSTATE unsigned int
reserved IT state

ArchMsg.Warning.tlb_contents_unknown

Translation has been enabled without invalidating the TLB. Fields:

EL unsigned int
Exception level of translation regime
INVALIDITY unsigned int
Which TLB may be invalid (if TLBs are separate)
NON_SECURE unsigned int
Security state of translation regime
VMID unsigned int
Current VMID, if applicable

ArchMsg.Warning.unknown_DLR_DSPSR

Exiting debug state with unknown DLR or DSPSR.

ArchMsg.Warning.unknown_ELR_SPSR

Returning from debug exception with unknown ELR or SPSR.

ArchMsg.Warning.unpredictable_a32_breakpoint

Fields:

ADDR1 unsigned int
Address
ADDR2 unsigned int
Breakpoint address
BAS unsigned int
BAS Field
IS_ADDRESS_MISMATCH_BREAKPOINT bool
Is Mismatch breakpoint, not a match breakpoint
N unsigned int
Breakpoint number
OFFSET unsigned int
Indicates which halfword actually considering into the A32 instruction

ArchMsg.Warning.unpredictable_banked_register_access

Fields:

R unsigned int
R bit
SYSM unsigned int
SYSm

ArchMsg.Warning.unpredictable_ccfail_undef

Fields:

OPCODE unsigned int
opcode of the instruction

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VADDR unsigned int
Virtual address of the instruction being executed

ArchMsg.Warning.unpredictable_ici_bits

ICI bits do not apply or are architecturally unpredictable for instruction.

ArchMsg.Warning.unpredictable_t32_breakpoint

Fields:

ADDR1 unsigned int
Address
ADDR2 unsigned int
Breakpoint address
BAS unsigned int
BAS Field
IS_ADDRESS_MISMATCH_BREAKPOINT bool
Is Mismatch breakpoint, not a match breakpoint
N unsigned int
Breakpoint number
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_t32_vector_catch

Fields:

ADDR unsigned int
Address
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_watchpoint_far

Fields:

LOWER_BOUND unsigned int
the lowest address accessed by the instruction that triggered the watchpoint
UPPER_BOUND unsigned int
the highest watchpointed address accessed by that instruction

ArchMsg.Warning.warning_AdvSIMDExpandImmUnexpectedZero

AdvSIMDExpandImm may treat this immediate value as UNPREDICTABLE.

ArchMsg.Warning.warning_ConditionalSMC

SMC instruction has UNPREDICTABLE effects when conditional when combined with traps.

ArchMsg.Warning.warning_access_crosses_page_boundary

Access crosses page boundary.

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault_on_both_pages

Access crosses page boundary has fault(s).

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_access_crosses_page_boundary_spanning_different_memory

Fields:

ADDR unsigned int
: address of access crossing page

MEMTYPE_PAGE1 signed int
: First Page's memory type (0-Normal, 1-Non-Normal)

MEMTYPE_PAGE2 signed int
: Second Page's memory type (0-Normal, 1-Non-Normal)

ArchMsg.Warning.warning_access_wraps_around_memory

Fields:

WIDTH unsigned int
: Address width

ArchMsg.Warning.warning_bx_from_thumbee

Fields:

ADDR unsigned int
: Destination address

ArchMsg.Warning.warning_ccsidr_unimplemented_level

Fields:

CSSELR unsigned int
: current effective value of CSSELR

ArchMsg.Warning.warning_change_to_ns_when_tge_set

Attempting to change NS to 1 when HCR.TGE = 1.

ArchMsg.Warning.warning_contiguous_bit_check_abort

Fields:

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ArchMsg.Warning.warning_contiguous_bit_error

Fields:

CONTIG_ADDR unsigned int
: address of the TLB first read, with which this entry is expected to be contiguous

CONTIG_DATA unsigned int
: expected contents based on the entry at CONTIG_ADDR

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ENTRY_DATA unsigned int
: contents of conflicting TLB entry

ArchMsg.Warning.warning_cp10_cp11_mismatch

Fields:

EL unsigned int
: EL owning the control

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_cp10_cp11_reserved_value

Fields:

CP10 unsigned int
: CP10 access

CP11 unsigned int
: CP11 access

ArchMsg.Warning.warning_csselr_level_out_of_range

Fields:

IMPLEMENTED unsigned int
: number of cache levels implemented

LEVEL unsigned int
: written value of CSSELR.Level

TYPE enum
: The type of cache selected by CSSELR.InD and TnD

ArchMsg.Warning.warning_debug_flow_control_bits_not_obeyed

Fields:

ERROR enum
: type of error resulting

REGISTER enum
: register under consideration

ArchMsg.Warning.warning_debug_register_access_during_reset

Fields:

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

ArchMsg.Warning.warning_decode_cps_inconsistent_fields

The mode setting fields in the CPS instruction are inconsistent.

ArchMsg.Warning.warning_decode_invalid_state

Instruction is unpredictable in current exception-level/security state.

ArchMsg.Warning.warning_default_cacheable_mmu_on

Default cacheable is enabled (HCR.DC = 1) while MMU is enabled.

ArchMsg.Warning.warning_default_cacheable_vmmu_off

Default cacheable is enabled (HCR.DC = 1) while HCR.VM = 0.

ArchMsg.Warning.warning_deprecated_wvr_bit_2_set

Fields:

N unsigned int
: index

WVR unsigned int
: DBGWVR

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_eret_while_software_step_active_pending

Missing ISB between setting MDSCR_EL1.SS and ERET.

ArchMsg.Warning.warning_exclusive_to_non_normal

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_exclusive_to_non_writeback

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_execute_from_device_memory

An attempt was made to execute from Device memory.

ArchMsg.Warning.warning_illegal_cpsr_mode

Fields:

NEW_MODE unsigned int
: New value of CPSR.M

ArchMsg.Warning.warning_illegal_srs_mode

Fields:

MODE unsigned int
: mode for Banked SP

ArchMsg.Warning.warning_implementation_defined_read_debug_register_in_SCS

Read Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_implementation_defined_sequential_security_transitions_supported

The behaviour of a sequential instruction fetches that cross from non-secure to secure memory and contain SG instruction is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_implementation_defined_stack_limit_check_supported

It is IMPLEMENTATION DEFINED whether stack pointer limit checking is performed for the this instructions. Fields:

ADDRESS unsigned int
: Address of instruction

OPCODE unsigned int
: instruction opcode

ArchMsg.Warning.warning_implementation_defined_write_debug_register_in_SCS

Write Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

DATA unsigned int
: data attempted to be written

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PPB_OFFSET unsigned int
debug register offset

ArchMsg.Warning.warning_invalid_tcr_granule

Fields:

REQUEST signed int
page size requested (or 0 for reserved)
SUBSTITUTE signed int
best guess available page size
TG_ID bool
bits TG0 or TG1

ArchMsg.Warning.warning_load_multiple_user_registers_from_user_mode

An LDM(user registers) instruction executed from user mode.

ArchMsg.Warning.warning_load_pc_from_unaligned

PC loaded from an unaligned location.

ArchMsg.Warning.warning_mte_in_mem_attr_not_supported

Fields:

ATTR unsigned int
Memory attribute

ArchMsg.Warning.warning_reserved_breakpoint_state_match

Fields:

IS_BREAKPOINT unsigned int
Is breakpoint, not watchpoint

ArchMsg.Warning.warning_shareability

Unpredictable: combination of the 1st and 2nd stages of translation is Normal Inner Non-Cacheable, Outer Non-Cacheable.

ArchMsg.Warning.warning_software_step_set_while_enabled

MDSCR_EL1.SS set to 1 while software step debug exceptions are enabled.

ArchMsg.Warning.warning_thumb_instruction_wraps_around_memory

Thumb instruction warps around memory.

ArchMsg.Warning.warning_ttbr_sbz_bits_are_not_zero

Fields:

MSB unsigned int
x-1
TTBR enum
which TTBR register

ArchMsg.Warning.warning_unaligned_address_dbgdtrrx_write

Fields:

ADDR unsigned int
Address

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unaligned_address_dbgdtrtx_read

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_unaligned_to_device

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_unaligned_to_strongly_ordered

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_unknown_sau_rnr

SAU_RNR was set to an unsupported value.

ArchMsg.Warning.warning_unpredictable_AIRCR_PRIP_and_BFHFNMINP

The effect of setting both AIRCR.BFHFNMINP and AIRCR.PRIP to 1 is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTCLRACTIVE_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTCLRACTIVE if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_and_SYSRESETREQ

When the processor is halted in Debug state, if a write to the register writes a 1 to both VECTRESET and SYSRESETREQ, the behavior is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTRESET if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_incorrect_VKEY

The value 0x05FA must be written to AIRCR.VKEY, otherwise the register write is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_EXE_RETURN_Reserved_Bit

EXC_RETURN[23:7] are reserved with the special condition that all bits should be written as one. Values other than all 1s are UNPREDICTABLE. Fields:

EXC_RETURN unsigned int
: Special PC value

SBO_MASK unsigned int
: Which bits Should Be One to not UNPRED

ArchMsg.Warning.warning_unpredictable_change_to_DEMCR_MON_STEP_at_insufficient_priority

The effect of changing DEMCR.MON_STEP at an execution priority that is lower than the priority of the DebugMonitor exception is CONSTRAINED UNPREDICTABLE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_change_to_priority_of_active_exception

Changing the priority of an active exception is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_clear_lspact

Fields:

DRVEXC_TAKEN unsigned int
: derived exception taken not pended

REACHED_LAST_STORE unsigned int
: only the last faulted

ArchMsg.Warning.warning_unpredictable_dbg_exit_unaligned_dlr

Debug exit to AArch32 with DLR[0] set to 1. PC[0] can be set to 0 or to DLR[0].

ArchMsg.Warning.warning_unpredictable_exception_catch

Generation of an exception catch event is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_inconsistent_state

state on exception return is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_instruction

exception return instruction is unpredictable.

ArchMsg.Warning.warning_unpredictable_in_debug_state

Instruction is UNPREDICTABLE when executed in debug state.

ArchMsg.Warning.warning_unpredictable_mair_encoding

Fields:

ATTR unsigned int
: Memory attribute

ArchMsg.Warning.warning_unpredictable_pmu_counter_access

Access to PMU counters from non-secure EL0 or EL1 is UNPREDICTABLE with MDCR_EL2.HPMN set to 0.

ArchMsg.Warning.warning_unpredictable_prioritization_breakpoint_match_vector_catch

Two events with the same priority occurred at the same instruction: Address Matching Vector Catch debug event, and Breakpoint debug event It is constrained unpredictable which is taken.

ArchMsg.Warning.warning_unpredictable_stack_selection

UNPREDICTABLE state on context switch.

ArchMsg.Warning.warning_unpredictable_tsize_out_of_range

Fields:

TCR enum
: Which TCR

TSIZE signed int
: Value of TSize

TSIZE_MAX signed int
: The maximum allowed value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

TTBR unsigned int
: Which TTBR region

ArchMsg.Warning.warning_unpredictable_unaligned_pc_as_base_register

The PC value must be word-aligned, otherwise the behavior of the instruction is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_unaligned_pop_stack

Fields:

FRAME_ADDR unsigned int
: unaligned stack frame address

ArchMsg.Warning.warning_unpredictable_vmsa_memattrib

Fields:

TEXCB unsigned int
: texcb value

TEX_REMAP bool
: use tex remap

ArchMsg.Warning.warning_unpredictable_vtcr_t0sz_sl0

Fields:

SL0 signed int
: VTCR.SL0, starting level for stage 2 translation table walks

T0SZ signed int
: VTCR.T0SZ, required input address range

ArchMsg.Warning.warning_unpredictable_write_DHCSR

Fields:

NEW unsigned int
: data attempted to be written

OLD unsigned int
: previous value

ArchMsg.Warning.warning_unpredictable_write_SHCSR

UNPREDICTABLE setting a pending bit to 1 for an exception with priority >= execution priority.

ArchMsg.Warning.warning_unsupported_access_to_memory_mapped_register

Access to memory mapped register is unsupported.

ArchMsg.Warning.warning_user_jmcr_access

Fields:

WRITE unsigned int
: access is write

ArchMsg.Warning.warning_wcr_mask_and_bas

DBGWCR_EL1.MASK is non-zero and BAS is not 0xff.

ArchMsg.Warning.warning_wcr_mask_reserved

DBGWCR_ELn.MASK is set to a reserved value.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_wcr_non_configuous_bas

DBGWCR_EL1.BAS has non-contiguous set of ones.

ArchMsg.Warning.warning_wvr_masked_bit_not_zero

DBGWVRn_EL1.VA contains a bit masked by DBGWCRn_EL1.MASK that is not zero.

ArchMsg.Why.why_branch_target_exception

A branch target exception has occurred. Fields:

BTYPE unsigned int
: pstate.BTYPE at time branch taken

OPCODE unsigned int
: opcode at branch target

PC unsigned int
: Virtual Address of branch target

ArchMsg.Why.why_illegal_state

An illegal mode change has occured. Fields:

MESSAGE string
: The message

BRANCH_MISPREDICT

Simulating branch mispredict. Fields:

PC unsigned int
: Origin address (or 0 if unavailable)

BRA_DIR

Direct branches, to immediate address. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR

Indirect branches, perhaps to a register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR_CCFAIL

Indirect branches fail, perhaps to a register.

BRA_RET_CCFAIL

Branche return fail, perhaps to a register.

CACHE_MAINTENANCE_OP

Cache Maintenance Operation. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Specified MVA or Set/way

FUNCTION enum
: Clean / Invalidate

SCOPE enum
: Affected region

SIDE enum
: Inst / Data.

CCFAIL

Conditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCFAIL_UNC

Unconditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS

Conditional instruction condition check pass. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS_UNC

Unconditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CHECKPOINT_MESSAGE

Report error messages from the checkpointing process. Fields:

message string
: Message contents.

CHECKPOINT_RESTORE_END

Checkpoint restore completed.

CHECKPOINT_RESTORE_START

Checkpoint restore about to start.

CHECKPOINT_SAVE_END

Checkpoint save completed.

CHECKPOINT_SAVE_START

Checkpoint save about to start.

CODE_CACHE_MAINT

Code cache maintenance. Fields:

END_INCL unsigned int
: Inclusive end addres for by VA or PA requests

NS bool
: Non-secure world for PA

START unsigned int
: Start address for by VA or PA requests

TYPE enum
: Request type.

COMPILE_BLOCK_END

Last instruction of basic block translated. Fields:

VADDR unsigned int
: Address of next instruction after this basic block

COMPILE_BLOCK_START

First instruction of basic block translated. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VADDR unsigned int
: Address of first instruction in basic block

COMPILE_INST

ARM instruction compiled. Fields:

DISASS string
: Disassembly of the instruction.

ISET enum
: The instruction set of this instruction.

ITSTATE unsigned int
: The ITSTATE current for the instruction.

OPCODE unsigned int
: The opcode of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

CONTEXTIDR

Context ID Register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

NS bool
: Secure or nonsecure banked register is accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

CONTEXT_SYNC

Called for every context synchronization event. Fields:

ARCH_CSE bool
: Architectural Context Synchronization Event

SYSREG_SYNC bool
: System register synchronization event

CORE_ENDIAN

Core BE8 Big-Endian state changed. Fields:

BE8 bool
: Core BE8 Big-Endian state

CORE_INFO

Static processor attributes. Only triggered by a call to DumpState(). Fields:

ARCH_PROFILE enum
: The architecture profile of the core.

CLUSTER_ID unsigned int
: The cluster ID of this processor.

CORE_NUM unsigned int
: The number of this core in an MP processor.

DETAILED_CORE_FEATURES enum
: Detailed core features as a bitfield. Enum values are bitmasks to extract the info from the field

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

FPU_VERSION enum
: The VFP version implemented by the core.

MEM_ARCH enum
: The memory architecture of the core.

MIDR_EL1 unsigned int
: MIDR_EL1 value

NUM_CORES unsigned int
: The number of cores in this MP processor.

QUANTUM_SIZE unsigned int
: The default quantum size of the core.

SECURITY_FEATURES bool
: Does the core have security features?

CORE_LOADS

Processor load accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_REGS

Changes of the core registers R0 to R14. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number, 0 to 14.

MODE enum
: Bank of the register accessed.

OLD_VALUE unsigned int
: The old value overwritten.

PHYS_ID enum
: The physical register accessed.

VALUE unsigned int
: The new value written to the register.

CORE_REGS64

Changes of the core registers X0..X30, SP_ELn. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: The register number, 0..30 for X0..X30, >=32 for SP_ELn.

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

CORE_STORES

Processor store accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CP14_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP14_WRITE

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

CP15_WRITE

System Control Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

CPSR

CPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

OLD_VALUE unsigned int
: The old CPSR value

UNKNOWN unsigned int
: Bits within the register that have unknown value.

VALUE unsigned int
: The new CPSR value

CRYPTO_SPEC

Every crypto instruction speculatively executed.

CorePowerStateChange

Core power-up state has changed. Fields:

PU bool
: Whether the core powered up

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DATA_CACHE_SET_ALLOC_TAGS

Set allocation tags from DC GVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of memory in bytes.

TAG unsigned int
: The value of the tag being set.

VADDR unsigned int
: Virtual address used for setting allocation tags.

DATA_CACHE_ZERO

Zero and invalidate cache line from DC ZVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of zero'd memory in bytes.

VADDR unsigned int
: Virtual address of the zero'd cache line.

DEBUG_EVENT

Hardware debug support event. Fields:

EVENT enum
: Description of event

VALUE unsigned int
: data value

END_COMPILE

Compilation end. Fields:

END_OF_PAGE_COUNT unsigned int
: Number of basic blocks exited due to page boundary since START_COMPILE.

FETCHFAIL_COUNT unsigned int
: Number of basic blocks exited due to ifetch failure START_COMPILE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: Number of instructions compiled since START_COMPILE.

NONSEQ_COUNT unsigned int
: Number of basic blocks exited due to non-sequential instructions since START_COMPILE.

PAGE_STRADDLE_COUNT unsigned int
: Number of basic blocks exited due to unaligned instructions crossing page since START_COMPILE.

ESB

Error Synchronization Barrier hit. Fields:

ASYNC bool
: ESB is async

IMPLICIT bool
: ESB is implicit

EXCEPTION

Exception taken. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

IS_PHYSICAL bool
: Physical or Virtual exception

LR unsigned int
: The value assigned to the link register.

PC unsigned int
: The location where the exception occurred.

PREFERRED_RETURN unsigned int
: The preferred return address for the exception.

TARGET_ISET enum
: The instruction set of the exception handler code.

TARGET_PC unsigned int
: The address the exception branches to.

VECTOR enum
: The exception vector.

VECTOR_OFFSET unsigned int
: The offset of an exception vector from the base address.

EXCEPTION_END

Every exception completed.

EXCEPTION_RAISE

Every exception raised.

EXCEPTION_RETURN

Branches on leaving exception. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

EXCEPTION_RETURN_PREBRANCH

Exception return event that must occur before PC and processor state have updated.

EXCEPTION_START

Every exception started.

FIQ_TAKEN

FIQ taken exception.

FP_STATE

Floating point state. Fields:

FPSR unsigned int
: FPSR register value.

MASK unsigned int
: The mask indicating updated cumulative bits.

HLT

HLT instruction occurred. Fields:

IMM16 unsigned int
: The 16-bit immediate value encoded in HLT instruction.

INST

Every instruction executed. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DEBUG_STATE bool
: The instruction is executed in debug state.

DISASS string
: Disassembly of instruction.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The current instruction set.

ITSTATE unsigned int
: The current ITSTATE.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

MODE enum
: The mode the core is in.

NS unsigned int
: The core's non-secure bit.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

OPCODE unsigned int
: The opcode of the instruction.

PADDR unsigned int
: The physical address of the instruction.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

INST_END

Every instruction completed.

INST_START

Every instruction started. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

INST_STRADDLE

Instruction straddles boundary. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

IRQ_TAKEN

IRQ taken exception.

ISB

ISB instruction occurred.

LOAD_MULTIPLE_REGS_LIST

List of destination registers (not source) of a multiple registers load instruction. Fields:

REG string
: Register to add into the list

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOCAL_MONITOR

Local monitor activity. Fields:

PADDR unsigned int
: Local Monitor Address

State enum
: State of the monitor (Open/Exclusive)

MEMMAP_DEBUG_READ

Memory mapped reads to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

MEMMAP_DEBUG_WRITE

Memory mapped writes to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

MMU_TRANS

Address translation information. Fields:

ASID unsigned int
: Address space identifier.

CORE_NUM unsigned int
: Core number in a multi processor.

Hyp bool
: Entry matches in Hyp state only

INNERCACHE_RA bool
: Is the inner cache allocate on read

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INNERCACHE_TRANSIENT bool
: Is the inner write-through transient

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

MEMTYPE enum
: Memory type.

NSDESC enum
: Is secure side supposed to access secure or nonsecure memory

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TRANSIENT bool
: Is the outer write-through transient

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

PADDR unsigned int
: Physical address of the access.

PAGESIZE unsigned int
: Page size as log2(size).

SH enum
: Shareability

SIDE enum
: Inst / Data.

STAGE1_PERM unsigned int
: Stage 1 permissions, [5:3] Privileged access (XWR) [2:0] User access (XWR)

STAGE2_PERM unsigned int
: Stage 2 permission mask [5:3] Privileged access (XWR) [2:0] User access (XWR)

VADDR unsigned int
: Virtual address of the access.

VMID unsigned int
: Virtual machine identifier.

XS bool
: XS Attribute

nG enum
: Flag indicating whether ASID will be matched

MMU_TTB_READ

This event is triggered by reads caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data read.

IPA unsigned int
: For stage 1, the IPA of the read.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OUTADDR unsigned int
: The output physical address.

PADDR unsigned int
: The physical address of the read.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MMU_TTB_WRITE

This event is triggered by writes caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data written.

IPA unsigned int
: For stage 1, the IPA of the write.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

PADDR unsigned int
: The physical address of the write.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MODE_CHANGE

Mode change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: The new mode.

NON_SECURE enum
: The new security state

OLD_MODE enum
: The old mode.

NOT_TAG_CHECKED_INST

MTE Tag not checked for current instruction.

PERIODIC

Called for every quantum. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PC unsigned int
The address of the next instruction to be executed on this CPU.

PMU_COUNTER_OVERFLOW

PMU counter overflow. Fields:

EVENT_ID unsigned int
Event Identifier
INDEX unsigned int
Counter index (as selected by PMSELR).
INTERRUPT bool
Is interrupt enabled on overflow for this counter

PREFETCH_MEMORY64

Prefetch from PRFM or PRFUM instructions. Fields:

PRFOP unsigned int
Prefetch hint
VADDR unsigned int
Virtual address of the location that should be prefetched.

PRELOAD_DATA

Data preload from PLD instruction. Fields:

VADDR unsigned int
Virtual address of the data that should be preloaded.

PRELOAD_INST

Instruction preload from PLI instruction. Fields:

VADDR unsigned int
Virtual address of the instruction that should be preloaded.

PRE_CORE_LOAD

Trace just before a core load. Fields:

CURRENT_TIME unsigned int
The core's current time, as simulated time plus local time.
LOCAL_TIME unsigned int
The core's local time, relative to the current quantum.

PRE_CORE_STORE

Trace just before a core store. Fields:

CURRENT_TIME unsigned int
The core's current time, as simulated time plus local time.
LOCAL_TIME unsigned int
The core's local time, relative to the current quantum.

PRE_TTB_READ

This event is triggered before reads caused by a translation table walk. Fields:

ADDR unsigned int
The physical address of the read.

PSTATE_PACTIVE_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions on PChannel. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

PSTATE_PACTIVE_OPMODE_TRANSITION_DENIED

Trace the denied pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_POWER_TRANSITION

Trace pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

PSTATE_PACTIVE_POWER_TRANSITION_DENIED

Trace the denied pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_THREAD_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions for threads on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

RUN_STATE

Run state transition. Fields:

INST_COUNT unsigned int
: Ticks count at point of transition.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NEW enum
: New run state.

OLD enum
: Old run state.

SEMIHOSTING_CALL

Call of a semihost function occurred. Fields:

PC unsigned int
: The program counter after the semihosting call.

SEMIHOSTING_PRECALL

About to call semihost. Fields:

EL2 bool
: Use EL2 translation regime.

NS bool
: Is Non-Secure.

REG_WIDTH unsigned int
: The current register width in bytes.

SIGNAL

External signal state change. Fields:

SIGNAL enum
: Signal that changed

STATE bool
: Signal asserted state

SOFTWARE_STEP

Return the Debug Software Step state. Fields:

STATE enum
: Software Step state

SPSR

SPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: Which of the banked SPSR registers is written.

OLD_VALUE unsigned int
: The old SPSR value

VALUE unsigned int
: The new SPSR value

START_COMPILE

Compilation started. Fields:

VADDR unsigned int
: Instruction where compilation begins.

STG_MULTIPLE_MICRO_OPS_ADDRESS_RANGE

Address range for MTE stores with multiple uops. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

STORE_MULTIPLE_REGS_ADDRESS_RANGE

Address range for stores with multiple registers. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

SUBOPTIMAL_LDST_RETIRED

suboptimal load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1: half-word, 2: word ...)

OPTIMAL_ALIGN_SELECTOR unsigned int
: Selector to decide the optimal alignment size (1: access width, 2: 4-bytes, 3: 8-bytes, ..., 12: 4-kbytes)

VADDR unsigned int
: The virtual address of the access

SYNC

Called for every synchronization, that is every quantum break. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

LOCAL_QUANTUM unsigned int
: The local quantum of this CPU.

LOCAL_TIME unsigned int
: The local time of this CPU.

SYNC_REASON enum
: Reason for sync.

SYSCALL

System call instruction executed. Fields:

IMM unsigned int
: Immediate value of the system call instruction.

TYPE enum
: System call type.

VADDR unsigned int
: Instruction that caused the system call.

SYSREG_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE32

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE64

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

UNALIGNED_LDST_RETIRED

Processor unaligned load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1=>half-word, 2->word ...)

NUMBER_OF_BEATS unsigned int
: Number of accesses (beats) in this burst (Total size in bytes is ACCESS_SIZE * NUMBER_OF_BEATS)

VADDR unsigned int
: The virtual address of the access

VFP_D_REGS

VFP/NEON D 64 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_Q_REGS

VFP/NEON Q 128 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding byte in the register. For example, 0x1 would indicate a write to VALUE[7:0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_SYS_REGS

Writes to the VFP/NEON units system registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: Which VFP system register is written.

MASK unsigned int
: The mask indicating updated cumulative bits.

OLD_VALUE unsigned int
: The register's old value overwritten.

VALUE unsigned int
: The new value written to the VFP system register.

VFP_S_REGS

VFP/NEON S 32 bit register write. Fields:

ALIASING enum
: Type of register aliasing used.

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

WAYPOINT

Signals end of basic block execution. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: Origin instructions set

IS_COND bool
: Indicates if this is a conditional waypoint.

PC unsigned int
: Origin address (or 0 if unavailable)

TAKEN bool
: Indicates if this waypoint was taken.

TARGET unsigned int
: Destination address

TARGET_ISET enum
: Destination instructions set

WFE_END

WFE ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFE.

WFE_EVENT_REGISTER

WFE event register status: set/clear, reason. Fields:

INST_COUNT unsigned int
: Ticks count.

REASON enum
: Reason for set/clear. Only REASON==0 (Cleared by WFE) clears the bit, all other reasons set it.

WFE_IGNORED

WFE ignored. Fields:

EVENT bool
: This WFE was ignored because the event register was set.

INST_COUNT unsigned int
: Ticks count when ignoring WFE.

TRAPPED bool
: This WFE was trapped.

WFE_START

WFE entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFE.

WFI_END

WFI ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFI.

WFI_IGNORED

WFI ignored. Fields:

DISABLED bool
: This WFI was ignored because WFI is disabled.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: Ticks count when ignoring WFI.

REASON enum
: specifies reason why WFI trace was ignored

TRAPPED bool
: This WFI was trapped.

WFI_START

WFI entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFI.

WFI_WAKEUP

WFI wakeup. Fields:

INST_COUNT unsigned int
: Ticks count when WFI wakeup occurred.

REASON enum
: Reason for wakeup.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.9 ARM_Cortex-A35 - trace

This section describes the trace sources.

ASYNC_MEMORY_FAULT

Context ID Register write. Fields:

FAULT unsigned int
: Fault status in ESR format

PADDR unsigned int
: Physical Address (or 0 if unavailable)

VADDR signed int
: Virtual Address (or 0 if unavailable)

ATOMIC_END_ACCESS

Bus trace access for atomics. Fields:

ACCESS_FAIL bool
: Memory access failed

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

LOAD_VALUE unsigned int
: Loaded values

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ATOMIC_START_ACCESS

Bus trace access for atomics. Fields:

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

NSDESC unsigned int
: The security state of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ArchMsg.Error.exit_code

Fields:

EXITCODE signed int
: Exit code to be returned

KIND string
: Component kind that invoked the exit code trace

REASON string
: Reason why the exit code trace was invoked (optional)

ArchMsg.Warning.IgnoredWarmResetRequest

Event fires when software attempts to request warm reset through write to deprecated DBGPRCR.CWRR.

ArchMsg.Warning.TarmacReconstructor

Unexpected event happened during Tarmac reconstruction. Fields:

REASON string
: Reason

ArchMsg.Warning.Unpredictably Indexed PM Event Register

Fields:

IsDirect unsigned int
: Direct Access rather than indirect

N unsigned int
: Number of Accessible Registers

SEL unsigned int
: Selector

ArchMsg.Warning.branch_to_unaligned_address

Fields:

ADDR unsigned int
: Unaligned branch address

ArchMsg.Warning.cache_contents_unknown

A cache has been activated without being invalidated and may contain UNKNOWN content. Fields:

REGIME unsigned int
: Current translation regime

SIDE unsigned int
: 0 d-side cache or unified cache, 1 i-side

ArchMsg.Warning.dcimvac_matches_watchpoint

Watchpoints matching an AArch32 DCIMVA are implementation defined.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_fieldmismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_fracbitsnegative

VCVT Instruction is unpredictable with a negative number of fraction bits.

ArchMsg.Warning.decode_initblock

Instruction is unpredictable in an IT block.

ArchMsg.Warning.decode_initblocknotlast

Instruction is unpredictable in an IT block when not the last.

ArchMsg.Warning.decode_invalid_condition

Instruction is unpredictable when the condition is not AL.

ArchMsg.Warning.decode_invalidfieldcombination

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_invalidvalue

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_mem_hint_unallocated

There is no memory hint allocated to this bit pattern.

ArchMsg.Warning.decode_registermatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registermismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registeroutofrange

Float point instruction is unpredictable indexing beyond the end of the register bank.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_sbzsbo

Reserved bits in the instruction are not canonical.

ArchMsg.Warning.decode_transactiontoobig

Attempt to perform a memory transaction of more than 128 bytes.

ArchMsg.Warning.decode_unpred_other

Instruction is unpredictable.

ArchMsg.Warning.decode_unpreduseofpc

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13inreglist

Instruction is unpredictable with R13 in register-list.

ArchMsg.Warning.decode_useofd16orhigher

Attempt to use register D16 or higher where it is not allowed.

ArchMsg.Warning.decode_writebackandbaseinlist

Instruction is unpredictable with write back when the base register is in the transfer list.

ArchMsg.Warning.decode_zeroregistersinlist

Instruction is unpredictable when no registers are to be transfered.

ArchMsg.Warning.recursive_branch

An instruction is performing a branch that targets the same instruction. If this was intended then a WFI might be more effective. Fields:

PC unsigned int
: Address of instruction causing the branch

ArchMsg.Warning.recursive_exception

An instruction has generated an exception that targets the same instruction. Fields:

CPSR unsigned int
: Processor state

PC unsigned int
: Address of instruction causing the exception

TYPE enum
: Type of exception

ArchMsg.Warning.reentrant_vector_catch

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR unsigned int
vector address

ArchMsg.Warning.reserved_it_state

Fields:

ITSTATE unsigned int
reserved IT state

ArchMsg.Warning.tlb_contents_unknown

Translation has been enabled without invalidating the TLB. Fields:

EL unsigned int
Exception level of translation regime
INVALIDITY unsigned int
Which TLB may be invalid (if TLBs are separate)
NON_SECURE unsigned int
Security state of translation regime
VMID unsigned int
Current VMID, if applicable

ArchMsg.Warning.unknown_DLR_DSPSR

Exiting debug state with unknown DLR or DSPSR.

ArchMsg.Warning.unknown_ELR_SPSR

Returning from debug exception with unknown ELR or SPSR.

ArchMsg.Warning.unpredictable_a32_breakpoint

Fields:

ADDR1 unsigned int
Address
ADDR2 unsigned int
Breakpoint address
BAS unsigned int
BAS Field
IS_ADDRESS_MISMATCH_BREAKPOINT bool
Is Mismatch breakpoint, not a match breakpoint
N unsigned int
Breakpoint number
OFFSET unsigned int
Indicates which halfword actually considering into the A32 instruction

ArchMsg.Warning.unpredictable_banked_register_access

Fields:

R unsigned int
R bit
SYSM unsigned int
SYSm

ArchMsg.Warning.unpredictable_ccfail_undef

Fields:

OPCODE unsigned int
opcode of the instruction

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VADDR unsigned int
Virtual address of the instruction being executed

ArchMsg.Warning.unpredictable_ici_bits

ICI bits do not apply or are architecturally unpredictable for instruction.

ArchMsg.Warning.unpredictable_t32_breakpoint

Fields:

ADDR1 unsigned int
Address
ADDR2 unsigned int
Breakpoint address
BAS unsigned int
BAS Field
IS_ADDRESS_MISMATCH_BREAKPOINT bool
Is Mismatch breakpoint, not a match breakpoint
N unsigned int
Breakpoint number
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_t32_vector_catch

Fields:

ADDR unsigned int
Address
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_watchpoint_far

Fields:

LOWER_BOUND unsigned int
the lowest address accessed by the instruction that triggered the watchpoint
UPPER_BOUND unsigned int
the highest watchpointed address accessed by that instruction

ArchMsg.Warning.warning_AdvSIMDExpandImmUnexpectedZero

AdvSIMDExpandImm may treat this immediate value as UNPREDICTABLE.

ArchMsg.Warning.warning_ConditionalSMC

SMC instruction has UNPREDICTABLE effects when conditional when combined with traps.

ArchMsg.Warning.warning_access_crosses_page_boundary

Access crosses page boundary.

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault_on_both_pages

Access crosses page boundary has fault(s).

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_access_crosses_page_boundary_spanning_different_memory

Fields:

ADDR unsigned int
: address of access crossing page

MEMTYPE_PAGE1 signed int
: First Page's memory type (0-Normal, 1-Non-Normal)

MEMTYPE_PAGE2 signed int
: Second Page's memory type (0-Normal, 1-Non-Normal)

ArchMsg.Warning.warning_access_wraps_around_memory

Fields:

WIDTH unsigned int
: Address width

ArchMsg.Warning.warning_bx_from_thumbee

Fields:

ADDR unsigned int
: Destination address

ArchMsg.Warning.warning_ccsidr_unimplemented_level

Fields:

CSSELR unsigned int
: current effective value of CSSELR

ArchMsg.Warning.warning_change_to_ns_when_tge_set

Attempting to change NS to 1 when HCR.TGE = 1.

ArchMsg.Warning.warning_contiguous_bit_check_abort

Fields:

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ArchMsg.Warning.warning_contiguous_bit_error

Fields:

CONTIG_ADDR unsigned int
: address of the TLB first read, with which this entry is expected to be contiguous

CONTIG_DATA unsigned int
: expected contents based on the entry at CONTIG_ADDR

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ENTRY_DATA unsigned int
: contents of conflicting TLB entry

ArchMsg.Warning.warning_cp10_cp11_mismatch

Fields:

EL unsigned int
: EL owning the control

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_cp10_cp11_reserved_value

Fields:

CP10 unsigned int
: CP10 access

CP11 unsigned int
: CP11 access

ArchMsg.Warning.warning_csselr_level_out_of_range

Fields:

IMPLEMENTED unsigned int
: number of cache levels implemented

LEVEL unsigned int
: written value of CSSELR.Level

TYPE enum
: The type of cache selected by CSSELR.InD and TnD

ArchMsg.Warning.warning_debug_flow_control_bits_not_obeyed

Fields:

ERROR enum
: type of error resulting

REGISTER enum
: register under consideration

ArchMsg.Warning.warning_debug_register_access_during_reset

Fields:

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

ArchMsg.Warning.warning_decode_cps_inconsistent_fields

The mode setting fields in the CPS instruction are inconsistent.

ArchMsg.Warning.warning_decode_invalid_state

Instruction is unpredictable in current exception-level/security state.

ArchMsg.Warning.warning_default_cacheable_mmu_on

Default cacheable is enabled (HCR.DC = 1) while MMU is enabled.

ArchMsg.Warning.warning_default_cacheable_vmmu_off

Default cacheable is enabled (HCR.DC = 1) while HCR.VM = 0.

ArchMsg.Warning.warning_deprecated_wvr_bit_2_set

Fields:

N unsigned int
: index

WVR unsigned int
: DBGWVR

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_eret_while_software_step_active_pending

Missing ISB between setting MDSCR_EL1.SS and ERET.

ArchMsg.Warning.warning_exclusive_to_non_normal

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_exclusive_to_non_writeback

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_execute_from_device_memory

An attempt was made to execute from Device memory.

ArchMsg.Warning.warning_illegal_cpsr_mode

Fields:

NEW_MODE unsigned int
: New value of CPSR.M

ArchMsg.Warning.warning_illegal_srs_mode

Fields:

MODE unsigned int
: mode for Banked SP

ArchMsg.Warning.warning_implementation_defined_read_debug_register_in_SCS

Read Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_implementation_defined_sequential_security_transitions_supported

The behaviour of a sequential instruction fetches that cross from non-secure to secure memory and contain SG instruction is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_implementation_defined_stack_limit_check_supported

It is IMPLEMENTATION DEFINED whether stack pointer limit checking is performed for the this instructions. Fields:

ADDRESS unsigned int
: Address of instruction

OPCODE unsigned int
: instruction opcode

ArchMsg.Warning.warning_implementation_defined_write_debug_register_in_SCS

Write Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

DATA unsigned int
: data attempted to be written

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PPB_OFFSET unsigned int
debug register offset

ArchMsg.Warning.warning_invalid_tcr_granule

Fields:

REQUEST signed int
page size requested (or 0 for reserved)
SUBSTITUTE signed int
best guess available page size
TG_ID bool
bits TG0 or TG1

ArchMsg.Warning.warning_load_multiple_user_registers_from_user_mode

An LDM(user registers) instruction executed from user mode.

ArchMsg.Warning.warning_load_pc_from_unaligned

PC loaded from an unaligned location.

ArchMsg.Warning.warning_mte_in_mem_attr_not_supported

Fields:

ATTR unsigned int
Memory attribute

ArchMsg.Warning.warning_reserved_breakpoint_state_match

Fields:

IS_BREAKPOINT unsigned int
Is breakpoint, not watchpoint

ArchMsg.Warning.warning_shareability

Unpredictable: combination of the 1st and 2nd stages of translation is Normal Inner Non-Cacheable, Outer Non-Cacheable.

ArchMsg.Warning.warning_software_step_set_while_enabled

MDSCR_EL1.SS set to 1 while software step debug exceptions are enabled.

ArchMsg.Warning.warning_thumb_instruction_wraps_around_memory

Thumb instruction warps around memory.

ArchMsg.Warning.warning_ttbr_sbz_bits_are_not_zero

Fields:

MSB unsigned int
x-1
TTBR enum
which TTBR register

ArchMsg.Warning.warning_unaligned_address_dbgdtrrx_write

Fields:

ADDR unsigned int
Address

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unaligned_address_dbgdtrtx_read

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_unaligned_to_device

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_unaligned_to_strongly_ordered

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_unknown_sau_rnr

SAU_RNR was set to an unsupported value.

ArchMsg.Warning.warning_unpredictable_AIRCR_PRIP_and_BFHFNMINP

The effect of setting both AIRCR.BFHFNMINP and AIRCR.PRIP to 1 is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTCLRACTIVE_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTCLRACTIVE if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_and_SYSRESETREQ

When the processor is halted in Debug state, if a write to the register writes a 1 to both VECTRESET and SYSRESETREQ, the behavior is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTRESET if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_incorrect_VKEY

The value 0x05FA must be written to AIRCR.VKEY, otherwise the register write is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_EXE_RETURN_Reserved_Bit

EXC_RETURN[23:7] are reserved with the special condition that all bits should be written as one. Values other than all 1s are UNPREDICTABLE. Fields:

EXC_RETURN unsigned int
: Special PC value

SBO_MASK unsigned int
: Which bits Should Be One to not UNPRED

ArchMsg.Warning.warning_unpredictable_change_to_DEMCR_MON_STEP_at_insufficient_priority

The effect of changing DEMCR.MON_STEP at an execution priority that is lower than the priority of the DebugMonitor exception is CONSTRAINED UNPREDICTABLE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_change_to_priority_of_active_exception

Changing the priority of an active exception is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_clear_lspact

Fields:

DRVEXC_TAKEN unsigned int
: derived exception taken not pended

REACHED_LAST_STORE unsigned int
: only the last faulted

ArchMsg.Warning.warning_unpredictable_dbg_exit_unaligned_dlr

Debug exit to AArch32 with DLR[0] set to 1. PC[0] can be set to 0 or to DLR[0].

ArchMsg.Warning.warning_unpredictable_exception_catch

Generation of an exception catch event is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_inconsistent_state

state on exception return is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_instruction

exception return instruction is unpredictable.

ArchMsg.Warning.warning_unpredictable_in_debug_state

Instruction is UNPREDICTABLE when executed in debug state.

ArchMsg.Warning.warning_unpredictable_mair_encoding

Fields:

ATTR unsigned int
: Memory attribute

ArchMsg.Warning.warning_unpredictable_pmu_counter_access

Access to PMU counters from non-secure EL0 or EL1 is UNPREDICTABLE with MDCR_EL2.HPMN set to 0.

ArchMsg.Warning.warning_unpredictable_prioritization_breakpoint_match_vector_catch

Two events with the same priority occurred at the same instruction: Address Matching Vector Catch debug event, and Breakpoint debug event It is constrained unpredictable which is taken.

ArchMsg.Warning.warning_unpredictable_stack_selection

UNPREDICTABLE state on context switch.

ArchMsg.Warning.warning_unpredictable_tsize_out_of_range

Fields:

TCR enum
: Which TCR

TSIZE signed int
: Value of TSize

TSIZE_MAX signed int
: The maximum allowed value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

TTBR unsigned int
: Which TTBR region

ArchMsg.Warning.warning_unpredictable_unaligned_pc_as_base_register

The PC value must be word-aligned, otherwise the behavior of the instruction is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_unaligned_pop_stack

Fields:

FRAME_ADDR unsigned int
: unaligned stack frame address

ArchMsg.Warning.warning_unpredictable_vmsa_memattrib

Fields:

TEXCB unsigned int
: texcb value

TEX_REMAP bool
: use tex remap

ArchMsg.Warning.warning_unpredictable_vtcr_t0sz_sl0

Fields:

SL0 signed int
: VTCR.SL0, starting level for stage 2 translation table walks

T0SZ signed int
: VTCR.T0SZ, required input address range

ArchMsg.Warning.warning_unpredictable_write_DHCSR

Fields:

NEW unsigned int
: data attempted to be written

OLD unsigned int
: previous value

ArchMsg.Warning.warning_unpredictable_write_SHCSR

UNPREDICTABLE setting a pending bit to 1 for an exception with priority >= execution priority.

ArchMsg.Warning.warning_unsupported_access_to_memory_mapped_register

Access to memory mapped register is unsupported.

ArchMsg.Warning.warning_user_jmcr_access

Fields:

WRITE unsigned int
: access is write

ArchMsg.Warning.warning_wcr_mask_and_bas

DBGWCR_EL1.MASK is non-zero and BAS is not 0xff.

ArchMsg.Warning.warning_wcr_mask_reserved

DBGWCR_ELn.MASK is set to a reserved value.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_wcr_non_configuous_bas

DBGWCR_EL1.BAS has non-contiguous set of ones.

ArchMsg.Warning.warning_wvr_masked_bit_not_zero

DBGWVRn_EL1.VA contains a bit masked by DBGWCRn_EL1.MASK that is not zero.

ArchMsg.Why.why_branch_target_exception

A branch target exception has occurred. Fields:

BTYPE unsigned int
: pstate.BTYPE at time branch taken

OPCODE unsigned int
: opcode at branch target

PC unsigned int
: Virtual Address of branch target

ArchMsg.Why.why_illegal_state

An illegal mode change has occured. Fields:

MESSAGE string
: The message

BRANCH_MISPREDICT

Simulating branch mispredict. Fields:

PC unsigned int
: Origin address (or 0 if unavailable)

BRA_DIR

Direct branches, to immediate address. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR

Indirect branches, perhaps to a register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR_CCFAIL

Indirect branches fail, perhaps to a register.

BRA_RET_CCFAIL

Branche return fail, perhaps to a register.

CACHE_MAINTENANCE_OP

Cache Maintenance Operation. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Specified MVA or Set/way

FUNCTION enum
: Clean / Invalidate

SCOPE enum
: Affected region

SIDE enum
: Inst / Data.

CCFAIL

Conditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCFAIL_UNC

Unconditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS

Conditional instruction condition check pass. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS_UNC

Unconditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CHECKPOINT_MESSAGE

Report error messages from the checkpointing process. Fields:

message string
: Message contents.

CHECKPOINT_RESTORE_END

Checkpoint restore completed.

CHECKPOINT_RESTORE_START

Checkpoint restore about to start.

CHECKPOINT_SAVE_END

Checkpoint save completed.

CHECKPOINT_SAVE_START

Checkpoint save about to start.

CODE_CACHE_MAINT

Code cache maintenance. Fields:

END_INCL unsigned int
: Inclusive end addres for by VA or PA requests

NS bool
: Non-secure world for PA

START unsigned int
: Start address for by VA or PA requests

TYPE enum
: Request type.

COMPILE_BLOCK_END

Last instruction of basic block translated. Fields:

VADDR unsigned int
: Address of next instruction after this basic block

COMPILE_BLOCK_START

First instruction of basic block translated. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VADDR unsigned int
: Address of first instruction in basic block

COMPILE_INST

ARM instruction compiled. Fields:

DISASS string
: Disassembly of the instruction.

ISET enum
: The instruction set of this instruction.

ITSTATE unsigned int
: The ITSTATE current for the instruction.

OPCODE unsigned int
: The opcode of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

CONTEXTIDR

Context ID Register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

NS bool
: Secure or nonsecure banked register is accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

CONTEXT_SYNC

Called for every context synchronization event. Fields:

ARCH_CSE bool
: Architectural Context Synchronization Event

SYSREG_SYNC bool
: System register synchronization event

CORE_ENDIAN

Core BE8 Big-Endian state changed. Fields:

BE8 bool
: Core BE8 Big-Endian state

CORE_INFO

Static processor attributes. Only triggered by a call to DumpState(). Fields:

ARCH_PROFILE enum
: The architecture profile of the core.

CLUSTER_ID unsigned int
: The cluster ID of this processor.

CORE_NUM unsigned int
: The number of this core in an MP processor.

DETAILED_CORE_FEATURES enum
: Detailed core features as a bitfield. Enum values are bitmasks to extract the info from the field

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

FPU_VERSION enum
: The VFP version implemented by the core.

MEM_ARCH enum
: The memory architecture of the core.

MIDR_EL1 unsigned int
: MIDR_EL1 value

NUM_CORES unsigned int
: The number of cores in this MP processor.

QUANTUM_SIZE unsigned int
: The default quantum size of the core.

SECURITY_FEATURES bool
: Does the core have security features?

CORE_LOADS

Processor load accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_REGS

Changes of the core registers R0 to R14. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number, 0 to 14.

MODE enum
: Bank of the register accessed.

OLD_VALUE unsigned int
: The old value overwritten.

PHYS_ID enum
: The physical register accessed.

VALUE unsigned int
: The new value written to the register.

CORE_REGS64

Changes of the core registers X0..X30, SP_ELn. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: The register number, 0..30 for X0..X30, >=32 for SP_ELn.

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

CORE_STORES

Processor store accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CP14_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP14_WRITE

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

CP15_WRITE

System Control Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

CPSR

CPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

OLD_VALUE unsigned int
: The old CPSR value

UNKNOWN unsigned int
: Bits within the register that have unknown value.

VALUE unsigned int
: The new CPSR value

CRYPTO_SPEC

Every crypto instruction speculatively executed.

CorePowerStateChange

Core power-up state has changed. Fields:

PU bool
: Whether the core powered up

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DATA_CACHE_SET_ALLOC_TAGS

Set allocation tags from DC GVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of memory in bytes.

TAG unsigned int
: The value of the tag being set.

VADDR unsigned int
: Virtual address used for setting allocation tags.

DATA_CACHE_ZERO

Zero and invalidate cache line from DC ZVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of zero'd memory in bytes.

VADDR unsigned int
: Virtual address of the zero'd cache line.

DEBUG_EVENT

Hardware debug support event. Fields:

EVENT enum
: Description of event

VALUE unsigned int
: data value

END_COMPILE

Compilation end. Fields:

END_OF_PAGE_COUNT unsigned int
: Number of basic blocks exited due to page boundary since START_COMPILE.

FETCHFAIL_COUNT unsigned int
: Number of basic blocks exited due to ifetch failure START_COMPILE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: Number of instructions compiled since START_COMPILE.

NONSEQ_COUNT unsigned int
: Number of basic blocks exited due to non-sequential instructions since START_COMPILE.

PAGE_STRADDLE_COUNT unsigned int
: Number of basic blocks exited due to unaligned instructions crossing page since START_COMPILE.

ESB

Error Synchronization Barrier hit. Fields:

ASYNC bool
: ESB is async

IMPLICIT bool
: ESB is implicit

EXCEPTION

Exception taken. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

IS_PHYSICAL bool
: Physical or Virtual exception

LR unsigned int
: The value assigned to the link register.

PC unsigned int
: The location where the exception occurred.

PREFERRED_RETURN unsigned int
: The preferred return address for the exception.

TARGET_ISET enum
: The instruction set of the exception handler code.

TARGET_PC unsigned int
: The address the exception branches to.

VECTOR enum
: The exception vector.

VECTOR_OFFSET unsigned int
: The offset of an exception vector from the base address.

EXCEPTION_END

Every exception completed.

EXCEPTION_RAISE

Every exception raised.

EXCEPTION_RETURN

Branches on leaving exception. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

EXCEPTION_RETURN_PREBRANCH

Exception return event that must occur before PC and processor state have updated.

EXCEPTION_START

Every exception started.

FIQ_TAKEN

FIQ taken exception.

FP_STATE

Floating point state. Fields:

FPSR unsigned int
: FPSR register value.

MASK unsigned int
: The mask indicating updated cumulative bits.

HLT

HLT instruction occurred. Fields:

IMM16 unsigned int
: The 16-bit immediate value encoded in HLT instruction.

INST

Every instruction executed. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DEBUG_STATE bool
: The instruction is executed in debug state.

DISASS string
: Disassembly of instruction.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The current instruction set.

ITSTATE unsigned int
: The current ITSTATE.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

MODE enum
: The mode the core is in.

NS unsigned int
: The core's non-secure bit.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

OPCODE unsigned int
: The opcode of the instruction.

PADDR unsigned int
: The physical address of the instruction.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

INST_END

Every instruction completed.

INST_START

Every instruction started. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

INST_STRADDLE

Instruction straddles boundary. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

IRQ_TAKEN

IRQ taken exception.

ISB

ISB instruction occurred.

LOAD_MULTIPLE_REGS_LIST

List of destination registers (not source) of a multiple registers load instruction. Fields:

REG string
: Register to add into the list

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOCAL_MONITOR

Local monitor activity. Fields:

PADDR unsigned int
: Local Monitor Address

State enum
: State of the monitor (Open/Exclusive)

MEMMAP_DEBUG_READ

Memory mapped reads to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

MEMMAP_DEBUG_WRITE

Memory mapped writes to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

MMU_TRANS

Address translation information. Fields:

ASID unsigned int
: Address space identifier.

CORE_NUM unsigned int
: Core number in a multi processor.

Hyp bool
: Entry matches in Hyp state only

INNERCACHE_RA bool
: Is the inner cache allocate on read

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INNERCACHE_TRANSIENT bool
: Is the inner write-through transient

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

MEMTYPE enum
: Memory type.

NSDESC enum
: Is secure side supposed to access secure or nonsecure memory

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TRANSIENT bool
: Is the outer write-through transient

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

PADDR unsigned int
: Physical address of the access.

PAGESIZE unsigned int
: Page size as log2(size).

SH enum
: Shareability

SIDE enum
: Inst / Data.

STAGE1_PERM unsigned int
: Stage 1 permissions, [5:3] Privileged access (XWR) [2:0] User access (XWR)

STAGE2_PERM unsigned int
: Stage 2 permission mask [5:3] Privileged access (XWR) [2:0] User access (XWR)

VADDR unsigned int
: Virtual address of the access.

VMID unsigned int
: Virtual machine identifier.

XS bool
: XS Attribute

nG enum
: Flag indicating whether ASID will be matched

MMU_TTB_READ

This event is triggered by reads caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data read.

IPA unsigned int
: For stage 1, the IPA of the read.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OUTADDR unsigned int
: The output physical address.

PADDR unsigned int
: The physical address of the read.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MMU_TTB_WRITE

This event is triggered by writes caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data written.

IPA unsigned int
: For stage 1, the IPA of the write.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

PADDR unsigned int
: The physical address of the write.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MODE_CHANGE

Mode change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: The new mode.

NON_SECURE enum
: The new security state

OLD_MODE enum
: The old mode.

NOT_TAG_CHECKED_INST

MTE Tag not checked for current instruction.

PERIODIC

Called for every quantum. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PC unsigned int
The address of the next instruction to be executed on this CPU.

PMU_COUNTER_OVERFLOW

PMU counter overflow. Fields:

EVENT_ID unsigned int
Event Identifier
INDEX unsigned int
Counter index (as selected by PMSELR).
INTERRUPT bool
Is interrupt enabled on overflow for this counter

PREFETCH_MEMORY64

Prefetch from PRFM or PRFUM instructions. Fields:

PRFOP unsigned int
Prefetch hint
VADDR unsigned int
Virtual address of the location that should be prefetched.

PRELOAD_DATA

Data preload from PLD instruction. Fields:

VADDR unsigned int
Virtual address of the data that should be preloaded.

PRELOAD_INST

Instruction preload from PLI instruction. Fields:

VADDR unsigned int
Virtual address of the instruction that should be preloaded.

PRE_CORE_LOAD

Trace just before a core load. Fields:

CURRENT_TIME unsigned int
The core's current time, as simulated time plus local time.
LOCAL_TIME unsigned int
The core's local time, relative to the current quantum.

PRE_CORE_STORE

Trace just before a core store. Fields:

CURRENT_TIME unsigned int
The core's current time, as simulated time plus local time.
LOCAL_TIME unsigned int
The core's local time, relative to the current quantum.

PRE_TTB_READ

This event is triggered before reads caused by a translation table walk. Fields:

ADDR unsigned int
The physical address of the read.

PSTATE_PACTIVE_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions on PChannel. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

PSTATE_PACTIVE_OPMODE_TRANSITION_DENIED

Trace the denied pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_POWER_TRANSITION

Trace pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

PSTATE_PACTIVE_POWER_TRANSITION_DENIED

Trace the denied pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_THREAD_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions for threads on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

RUN_STATE

Run state transition. Fields:

INST_COUNT unsigned int
: Ticks count at point of transition.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NEW enum
: New run state.

OLD enum
: Old run state.

SEMIHOSTING_CALL

Call of a semihost function occurred. Fields:

PC unsigned int
: The program counter after the semihosting call.

SEMIHOSTING_PRECALL

About to call semihost. Fields:

EL2 bool
: Use EL2 translation regime.

NS bool
: Is Non-Secure.

REG_WIDTH unsigned int
: The current register width in bytes.

SIGNAL

External signal state change. Fields:

SIGNAL enum
: Signal that changed

STATE bool
: Signal asserted state

SOFTWARE_STEP

Return the Debug Software Step state. Fields:

STATE enum
: Software Step state

SPSR

SPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: Which of the banked SPSR registers is written.

OLD_VALUE unsigned int
: The old SPSR value

VALUE unsigned int
: The new SPSR value

START_COMPILE

Compilation started. Fields:

VADDR unsigned int
: Instruction where compilation begins.

STG_MULTIPLE_MICRO_OPS_ADDRESS_RANGE

Address range for MTE stores with multiple uops. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

STORE_MULTIPLE_REGS_ADDRESS_RANGE

Address range for stores with multiple registers. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

SUBOPTIMAL_LDST_RETIRED

suboptimal load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1: half-word, 2: word ...)

OPTIMAL_ALIGN_SELECTOR unsigned int
: Selector to decide the optimal alignment size (1: access width, 2: 4-bytes, 3: 8-bytes, ..., 12: 4-kbytes)

VADDR unsigned int
: The virtual address of the access

SYNC

Called for every synchronization, that is every quantum break. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

LOCAL_QUANTUM unsigned int
: The local quantum of this CPU.

LOCAL_TIME unsigned int
: The local time of this CPU.

SYNC_REASON enum
: Reason for sync.

SYSCALL

System call instruction executed. Fields:

IMM unsigned int
: Immediate value of the system call instruction.

TYPE enum
: System call type.

VADDR unsigned int
: Instruction that caused the system call.

SYSREG_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE32

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE64

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

UNALIGNED_LDST_RETIRED

Processor unaligned load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1=>half-word, 2->word ...)

NUMBER_OF_BEATS unsigned int
: Number of accesses (beats) in this burst (Total size in bytes is ACCESS_SIZE * NUMBER_OF_BEATS)

VADDR unsigned int
: The virtual address of the access

VFP_D_REGS

VFP/NEON D 64 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_Q_REGS

VFP/NEON Q 128 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding byte in the register. For example, 0x1 would indicate a write to VALUE[7:0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_SYS_REGS

Writes to the VFP/NEON units system registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: Which VFP system register is written.

MASK unsigned int
: The mask indicating updated cumulative bits.

OLD_VALUE unsigned int
: The register's old value overwritten.

VALUE unsigned int
: The new value written to the VFP system register.

VFP_S_REGS

VFP/NEON S 32 bit register write. Fields:

ALIASING enum
: Type of register aliasing used.

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

WAYPOINT

Signals end of basic block execution. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: Origin instructions set

IS_COND bool
: Indicates if this is a conditional waypoint.

PC unsigned int
: Origin address (or 0 if unavailable)

TAKEN bool
: Indicates if this waypoint was taken.

TARGET unsigned int
: Destination address

TARGET_ISET enum
: Destination instructions set

WFE_END

WFE ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFE.

WFE_EVENT_REGISTER

WFE event register status: set/clear, reason. Fields:

INST_COUNT unsigned int
: Ticks count.

REASON enum
: Reason for set/clear. Only REASON==0 (Cleared by WFE) clears the bit, all other reasons set it.

WFE_IGNORED

WFE ignored. Fields:

EVENT bool
: This WFE was ignored because the event register was set.

INST_COUNT unsigned int
: Ticks count when ignoring WFE.

TRAPPED bool
: This WFE was trapped.

WFE_START

WFE entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFE.

WFI_END

WFI ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFI.

WFI_IGNORED

WFI ignored. Fields:

DISABLED bool
: This WFI was ignored because WFI is disabled.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: Ticks count when ignoring WFI.

REASON enum
: specifies reason why WFI trace was ignored

TRAPPED bool
: This WFI was trapped.

WFI_START

WFI entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFI.

WFI_WAKEUP

WFI wakeup. Fields:

INST_COUNT unsigned int
: Ticks count when WFI wakeup occurred.

REASON enum
: Reason for wakeup.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.10 ARM_Cortex-A5 - trace

This section describes the trace sources.

ASYNC_MEMORY_FAULT

Context ID Register write. Fields:

FAULT unsigned int
: Fault status in ESR format

PADDR unsigned int
: Physical Address (or 0 if unavailable)

VADDR signed int
: Virtual Address (or 0 if unavailable)

ATOMIC_END_ACCESS

Bus trace access for atomics. Fields:

ACCESS_FAIL bool
: Memory access failed

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

LOAD_VALUE unsigned int
: Loaded values

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ATOMIC_START_ACCESS

Bus trace access for atomics. Fields:

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

NSDESC unsigned int
: The security state of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ArchMsg.Error.exit_code

Fields:

EXITCODE signed int
: Exit code to be returned

KIND string
: Component kind that invoked the exit code trace

REASON string
: Reason why the exit code trace was invoked (optional)

ArchMsg.Warning.TarmacReconstructor

Unexpected event happened during Tarmac reconstruction. Fields:

REASON string
: Reason

ArchMsg.Warning.cache_contents_unknown

A cache has been activated without being invalidated and may contain UNKNOWN content. Fields:

REGIME unsigned int
: Current translation regime

SIDE unsigned int
: 0 d-side cache or unified cache, 1 i-side

ArchMsg.Warning.recursive_branch

An instruction is performing a branch that targets the same instruction. If this was intended then a WFI might be more effective. Fields:

PC unsigned int
: Address of instruction causing the branch

ArchMsg.Warning.recursive_exception

An instruction has generated an exception that targets the same instruction. Fields:

CPSR unsigned int
: Processor state

PC unsigned int
: Address of instruction causing the exception

TYPE enum
: Type of exception

ArchMsg.Warning.tlb_contents_unknown

Translation has been enabled without invalidating the TLB. Fields:

EL unsigned int
: Exception level of translation regime

INVALIDITY unsigned int
: Which TLB may be invalid (if TLBs are separate)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NON_SECURE unsigned int
: Security state of translation regime

VMID unsigned int
: Current VMID, if applicable

BRANCH_MISPREDICT

Simulating branch mispredict. Fields:

PC unsigned int
: Origin address (or 0 if unavailable)

BRA_DIR

Direct branches, to immediate address. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR

Indirect branches, perhaps to a register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR_CCFAIL

Indirect branches fail, perhaps to a register.

BRA_RET_CCFAIL

Branche return fail, perhaps to a register.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CACHE_MAINTENANCE_OP

Cache Maintenance Operation. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Specified MVA or Set/way

FUNCTION enum
: Clean / Invalidate

SCOPE enum
: Affected region

SIDE enum
: Inst / Data.

CCFAIL

Conditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCFAIL_UNC

Unconditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS

Conditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS_UNC

Unconditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CODE_CACHE_MAINT

Code cache maintenance. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

END_INCL unsigned int
: Inclusive end addres for by VA or PA requests

NS bool
: Non-secure world for PA

START unsigned int
: Start address for by VA or PA requests

TYPE enum
: Request type.

COMPILE_BLOCK_END

Last instruction of basic block translated. Fields:

VADDR unsigned int
: Address of next instruction after this basic block

COMPILE_BLOCK_START

First instruction of basic block translated. Fields:

VADDR unsigned int
: Address of first instruction in basic block

COMPILE_INST

ARM instruction compiled. Fields:

DISASS string
: Disassembly of the instruction.

ISET enum
: The instruction set of this instruction.

ITSTATE unsigned int
: The ITSTATE current for the instruction.

OPCODE unsigned int
: The opcode of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

CONTEXTIDR

Context ID Register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

NS bool
: Secure or nonsecure banked register is accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

CONTEXT_SYNC

Called for every context synchronization event. Fields:

ARCH_CSE bool
: Architectural Context Synchronization Event

SYSREG_SYNC bool
: System register synchronization event

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_ENDIAN

Core BE8 Big-Endian state changed. Fields:

BE8 bool
: Core BE8 Big-Endian state

CORE_INFO

Static processor attributes. Only triggered by a call to DumpState(). Fields:

ARCH_PROFILE enum
: The architecture profile of the core.

CLUSTER_ID unsigned int
: The cluster ID of this processor.

CORE_NUM unsigned int
: The number of this core in an MP processor.

DETAILED_CORE_FEATURES enum
: Detailed core features as a bitfield. Enum values are bitmasks to extract the info from the field

FPU_VERSION enum
: The VFP version implemented by the core.

MEM_ARCH enum
: The memory architecture of the core.

MIDR_EL1 unsigned int
: MIDR_EL1 value

NUM_CORES unsigned int
: The number of cores in this MP processor.

QUANTUM_SIZE unsigned int
: The default quantum size of the core.

SECURITY_FEATURES bool
: Does the core have security features?

CORE_LOADS

Processor load accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CORE_REGS

Changes of the core registers R0 to R14. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number, 0 to 14.

MODE enum
: Bank of the register accessed.

OLD_VALUE unsigned int
: The old value overwritten.

PHYS_ID enum
: The physical register accessed.

VALUE unsigned int
: The new value written to the register.

CORE_STORES

Processor store accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CP14_READ

Reads to the System Coprocessor registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP14_WRITE

Writes to the System Coprocessor registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

CP15_WRITE

System Control Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

CPSR

CPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

OLD_VALUE unsigned int
: The old CPSR value

UNKNOWN unsigned int
: Bits within the register that have unknown value.

VALUE unsigned int
: The new CPSR value

CRYPTO_SPEC

Every crypto instruction speculatively executed.

DATA_CACHE_SET_ALLOC_TAGS

Set allocation tags from DC GVA instruction. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of memory in bytes.

TAG unsigned int
: The value of the tag being set.

VADDR unsigned int
: Virtual address used for setting allocation tags.

DATA_CACHE_ZERO

Zero and invalidate cache line from DC ZVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of zero'd memory in bytes.

VADDR unsigned int
: Virtual address of the zero'd cache line.

END_COMPILE

Compilation end. Fields:

END_OF_PAGE_COUNT unsigned int
: Number of basic blocks exited due to page boundary since START_COMPILE.

FETCHFAIL_COUNT unsigned int
: Number of basic blocks exited due to ifetch failure START_COMPILE.

INST_COUNT unsigned int
: Number of instructions compiled since START_COMPILE.

NONSEQ_COUNT unsigned int
: Number of basic blocks exited due to non-sequential instructions since START_COMPILE.

PAGE_STRADDLE_COUNT unsigned int
: Number of basic blocks exited due to unaligned instructions crossing page since START_COMPILE.

EXCEPTION

Exception taken. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

IS_PHYSICAL bool
: Physical or Virtual exception

LR unsigned int
: The value assigned to the link register.

PC unsigned int
: The location where the exception occurred.

PREFERRED_RETURN unsigned int
: The preferred return address for the exception.

TARGET_ISET enum
: The instruction set of the exception handler code.

TARGET_PC unsigned int
: The address the exception branches to.

VECTOR enum
: The exception vector.

VECTOR_OFFSET unsigned int
: The offset of an exception vector from the base address.

EXCEPTION_END

Every exception completed.

EXCEPTION_RAISE

Every exception raised.

EXCEPTION_RETURN

Branches on leaving exception. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

EXCEPTION_RETURN_PREBRANCH

Exception return event that must occur before PC and processor state have updated.

EXCEPTION_START

Every exception started.

FP_STATE

Floating point state. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

FPSR unsigned int
: FPSR register value.

MASK unsigned int
: The mask indicating updated cumulative bits.

HLT

HLT instruction occurred. Fields:

IMM16 unsigned int
: The 16-bit immediate value encoded in HLT instruction.

INST

Every instruction executed. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DEBUG_STATE bool
: The instruction is executed in debug state.

DISASS string
: Disassembly of instruction.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The current instruction set.

ITSTATE unsigned int
: The current ITSTATE.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

MODE enum
: The mode the core is in.

NS unsigned int
: The core's non-secure bit.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

OPCODE unsigned int
: The opcode of the instruction.

PADDR unsigned int
: The physical address of the instruction.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

INST_END

Every instruction completed.

INST_START

Every instruction started. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

INST_STRADDLE

Instruction straddles boundary. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

ISB

ISB instruction occurred.

LOAD_MULTIPLE_REGS_LIST

List of destination registers (not source) of a multiple registers load instruction. Fields:

REG string
: Register to add into the list

LOCAL_MONITOR

Local monitor activity. Fields:

PADDR unsigned int
: Local Monitor Address

State enum
: State of the monitor (Open/Exclusive)

MMU_TRANS

Address translation information. Fields:

ASID unsigned int
: Address space identifier.

CORE_NUM unsigned int
: Core number in a multi processor.

Hyp bool
: Entry matches in Hyp state only

INNERCACHE_RA bool
: Is the inner cache allocate on read

INNERCACHE_TRANSIENT bool
: Is the inner write-through transient

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MEMTYPE enum
: Memory type.

NSDESC enum
: Is secure side supposed to access secure or nonsecure memory

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TRANSIENT bool
: Is the outer write-through transient

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

PADDR unsigned int
: Physical address of the access.

PAGESIZE unsigned int
: Page size as log2(size).

SH enum
: Shareability

SIDE enum
: Inst / Data.

STAGE1_PERM unsigned int
: Stage 1 permissions, [5:3] Privileged access (XWR) [2:0] User access (XWR)

STAGE2_PERM unsigned int
: Stage 2 permission mask [5:3] Privileged access (XWR) [2:0] User access (XWR)

VADDR unsigned int
: Virtual address of the access.

VMID unsigned int
: Virtual machine identifier.

XS bool
: XS Attribute

nG enum
: Flag indicating whether ASID will be matched

MMU_TTB_READ

This event is triggered by reads caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data read.

IPA unsigned int
: For stage 1, the IPA of the read.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

OUTADDR unsigned int
: The output physical address.

PADDR unsigned int
: The physical address of the read.

SIDE enum
: Inst / Data.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

STAGE unsigned int
: Translation stage

MMU_TTB_WRITE

This event is triggered by writes caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data written.

IPA unsigned int
: For stage 1, the IPA of the write.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

PADDR unsigned int
: The physical address of the write.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MODE_CHANGE

Mode change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: The new mode.

NON_SECURE enum
: The new security state

OLD_MODE enum
: The old mode.

NOT_TAG_CHECKED_INST

MTE Tag not checked for current instruction.

PERIODIC

Called for every quantum. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

PC unsigned int
: The address of the next instruction to be executed on this CPU.

PMU_COUNTER_OVERFLOW

PMU counter overflow. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

EVENT_ID unsigned int
: Event Identifier

INDEX unsigned int
: Counter index (as selected by PMSELR).

INTERRUPT bool
: Is interrupt enabled on overflow for this counter

PRELOAD_DATA

Data preload from PLD instruction. Fields:

VADDR unsigned int
: Virtual address of the data that should be preloaded.

PRELOAD_INST

Instruction preload from PLI instruction. Fields:

VADDR unsigned int
: Virtual address of the instruction that should be preloaded.

PRE_CORE_LOAD

Trace just before a core load. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_CORE_STORE

Trace just before a core store. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_TTB_READ

This event is triggered before reads caused by a translation table walk. Fields:

ADDR unsigned int
: The physical address of the read.

PSTATE_PACTIVE_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

PSTATE_PACTIVE_OPMODE_TRANSITION_DENIED

Trace the denied pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_POWER_TRANSITION

Trace pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

PSTATE_PACTIVE_POWER_TRANSITION_DENIED

Trace the denied pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_THREAD_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions for threads on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

RUN_STATE

Run state transition. Fields:

INST_COUNT unsigned int
: Ticks count at point of transition.

NEW enum
: New run state.

OLD enum
: Old run state.

SEMIHOSTING_CALL

Call of a semihost function occurred. Fields:

PC unsigned int
: The program counter after the semihosting call.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SEMIHOSTING_PRECALL

About to call semihost. Fields:

EL2 bool
: Use EL2 translation regime.

NS bool
: Is Non-Secure.

REG_WIDTH unsigned int
: The current register width in bytes.

SIGNAL

External signal state change. Fields:

SIGNAL enum
: Signal that changed

STATE bool
: Signal asserted state

SPSR

SPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: Which of the banked SPSR registers is written.

OLD_VALUE unsigned int
: The old SPSR value

VALUE unsigned int
: The new SPSR value

START_COMPILE

Compilation started. Fields:

VADDR unsigned int
: Instruction where compilation begins.

STG_MULTIPLE_MICRO_OPS_ADDRESS_RANGE

Address range for MTE stores with multiple uops. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

STORE_MULTIPLE_REGS_ADDRESS_RANGE

Address range for stores with multiple registers. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

SUBOPTIMAL_LDST_RETIRED

suboptimal load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1: half-word, 2: word ...)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPTIMAL_ALIGN_SELECTOR unsigned int
: Selector to decide the optimal alignment size (1: access width, 2: 4-bytes, 3: 8-bytes, ..., 12: 4-kbytes)

VADDR unsigned int
: The virtual address of the access

SYNC

Called for every synchronization, that is every quantum break. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

LOCAL_QUANTUM unsigned int
: The local quantum of this CPU.

LOCAL_TIME unsigned int
: The local time of this CPU.

SYNC_REASON enum
: Reason for sync.

SYSCALL

System call instruction executed. Fields:

IMM unsigned int
: Immediate value of the system call instruction.

TYPE enum
: System call type.

VADDR unsigned int
: Instruction that caused the system call.

SYSREG_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SYSREG_UPDATE32

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE64

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

UNALIGNED_LDST_RETIRED

Processor unaligned load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1=>half-word, 2->word ...)

NUMBER_OF_BEATS unsigned int
: Number of accesses (beats) in this burst (Total size in bytes is ACCESS_SIZE * NUMBER_OF_BEATS)

VADDR unsigned int
: The virtual address of the access

VFP_D_REGS

VFP/NEON D 64 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_Q_REGS

VFP/NEON Q 128 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MASK unsigned int
: Each bit indicates a write to the corresponding byte in the register. For example, 0x1 would indicate a write to VALUE[7:0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_SYS_REGS

Writes to the VFP/NEON units system registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: Which VFP system register is written.

MASK unsigned int
: The mask indicating updated cumulative bits.

OLD_VALUE unsigned int
: The register's old value overwritten.

VALUE unsigned int
: The new value written to the VFP system register.

VFP_S_REGS

VFP/NEON S 32 bit register write. Fields:

ALIASING enum
: Type of register aliasing used.

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

WAYPOINT

Signals end of basic block execution. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: Origin instructions set

IS_COND bool
: Indicates if this is a conditional waypoint.

PC unsigned int
: Origin address (or 0 if unavailable)

TAKEN bool
: Indicates if this waypoint was taken.

TARGET unsigned int
: Destination address

TARGET_ISET enum
: Destination instructions set

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

WFE_END

WFE ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFE.

WFE_EVENT_REGISTER

WFE event register status: set/clear, reason. Fields:

INST_COUNT unsigned int
: Ticks count.

REASON enum
: Reason for set/clear. Only REASON==0 (Cleared by WFE) clears the bit, all other reasons set it.

WFE_IGNORED

WFE ignored. Fields:

EVENT bool
: This WFE was ignored because the event register was set.

INST_COUNT unsigned int
: Ticks count when ignoring WFE.

TRAPPED bool
: This WFE was trapped.

WFE_START

WFE entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFE.

WFI_END

WFI ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFI.

WFI_IGNORED

WFI ignored. Fields:

DISABLED bool
: This WFI was ignored because WFI is disabled.

INST_COUNT unsigned int
: Ticks count when ignoring WFI.

REASON enum
: specifies reason why WFI trace was ignored

TRAPPED bool
: This WFI was trapped.

WFI_START

WFI entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFI.

WFI_WAKEUP

WFI wakeup. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT `unsigned int`
: Ticks count when WFI wakeup occurred.

REASON `enum`
: Reason for wakeup.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.11 ARM_Cortex-A53 - trace

This section describes the trace sources.

ASYNC_MEMORY_FAULT

Context ID Register write. Fields:

FAULT unsigned int
: Fault status in ESR format

PADDR unsigned int
: Physical Address (or 0 if unavailable)

VADDR signed int
: Virtual Address (or 0 if unavailable)

ATOMIC_END_ACCESS

Bus trace access for atomics. Fields:

ACCESS_FAIL bool
: Memory access failed

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

LOAD_VALUE unsigned int
: Loaded values

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ATOMIC_START_ACCESS

Bus trace access for atomics. Fields:

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

NSDESC unsigned int
: The security state of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ArchMsg.Error.exit_code

Fields:

EXITCODE signed int
: Exit code to be returned

KIND string
: Component kind that invoked the exit code trace

REASON string
: Reason why the exit code trace was invoked (optional)

ArchMsg.Warning.IgnoredWarmResetRequest

Event fires when software attempts to request warm reset through write to deprecated DBGPRCR.CWRR.

ArchMsg.Warning.TarmacReconstructor

Unexpected event happened during Tarmac reconstruction. Fields:

REASON string
: Reason

ArchMsg.Warning.Unpredictably Indexed PM Event Register

Fields:

IsDirect unsigned int
: Direct Access rather than indirect

N unsigned int
: Number of Accessible Registers

SEL unsigned int
: Selector

ArchMsg.Warning.branch_to_unaligned_address

Fields:

ADDR unsigned int
: Unaligned branch address

ArchMsg.Warning.cache_contents_unknown

A cache has been activated without being invalidated and may contain UNKNOWN content. Fields:

REGIME unsigned int
: Current translation regime

SIDE unsigned int
: 0 d-side cache or unified cache, 1 i-side

ArchMsg.Warning.dcimvac_matches_watchpoint

Watchpoints matching an AArch32 DCIMVA are implementation defined.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_fieldmismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_fracbitsnegative

VCVT Instruction is unpredictable with a negative number of fraction bits.

ArchMsg.Warning.decode_initblock

Instruction is unpredictable in an IT block.

ArchMsg.Warning.decode_initblocknotlast

Instruction is unpredictable in an IT block when not the last.

ArchMsg.Warning.decode_invalid_condition

Instruction is unpredictable when the condition is not AL.

ArchMsg.Warning.decode_invalidfieldcombination

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_invalidvalue

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_mem_hint_unallocated

There is no memory hint allocated to this bit pattern.

ArchMsg.Warning.decode_registermatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registermismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registeroutofrange

Float point instruction is unpredictable indexing beyond the end of the register bank.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_sbzsbo

Reserved bits in the instruction are not canonical.

ArchMsg.Warning.decode_transactiontoobig

Attempt to perform a memory transaction of more than 128 bytes.

ArchMsg.Warning.decode_unpred_other

Instruction is unpredictable.

ArchMsg.Warning.decode_unpreduseofpc

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13inreglist

Instruction is unpredictable with R13 in register-list.

ArchMsg.Warning.decode_useofd16orhigher

Attempt to use register D16 or higher where it is not allowed.

ArchMsg.Warning.decode_writebackandbaseinlist

Instruction is unpredictable with write back when the base register is in the transfer list.

ArchMsg.Warning.decode_zeroregistersinlist

Instruction is unpredictable when no registers are to be transfered.

ArchMsg.Warning.recursive_branch

An instruction is performing a branch that targets the same instruction. If this was intended then a WFI might be more effective. Fields:

PC unsigned int
: Address of instruction causing the branch

ArchMsg.Warning.recursive_exception

An instruction has generated an exception that targets the same instruction. Fields:

CPSR unsigned int
: Processor state

PC unsigned int
: Address of instruction causing the exception

TYPE enum
: Type of exception

ArchMsg.Warning.reentrant_vector_catch

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR unsigned int
vector address

ArchMsg.Warning.reserved_it_state

Fields:

ITSTATE unsigned int
reserved IT state

ArchMsg.Warning.tlb_contents_unknown

Translation has been enabled without invalidating the TLB. Fields:

EL unsigned int
Exception level of translation regime
INVALIDITY unsigned int
Which TLB may be invalid (if TLBs are separate)
NON_SECURE unsigned int
Security state of translation regime
VMID unsigned int
Current VMID, if applicable

ArchMsg.Warning.unknown_DLR_DSPSR

Exiting debug state with unknown DLR or DSPSR.

ArchMsg.Warning.unknown_ELR_SPSR

Returning from debug exception with unknown ELR or SPSR.

ArchMsg.Warning.unpredictable_a32_breakpoint

Fields:

ADDR1 unsigned int
Address
ADDR2 unsigned int
Breakpoint address
BAS unsigned int
BAS Field
IS_ADDRESS_MISMATCH_BREAKPOINT bool
Is Mismatch breakpoint, not a match breakpoint
N unsigned int
Breakpoint number
OFFSET unsigned int
Indicates which halfword actually considering into the A32 instruction

ArchMsg.Warning.unpredictable_banked_register_access

Fields:

R unsigned int
R bit
SYSM unsigned int
SYSm

ArchMsg.Warning.unpredictable_ccfail_undef

Fields:

OPCODE unsigned int
opcode of the instruction

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VADDR unsigned int
Virtual address of the instruction being executed

ArchMsg.Warning.unpredictable_ici_bits

ICI bits do not apply or are architecturally unpredictable for instruction.

ArchMsg.Warning.unpredictable_t32_breakpoint

Fields:

ADDR1 unsigned int
Address
ADDR2 unsigned int
Breakpoint address
BAS unsigned int
BAS Field
IS_ADDRESS_MISMATCH_BREAKPOINT bool
Is Mismatch breakpoint, not a match breakpoint
N unsigned int
Breakpoint number
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_t32_vector_catch

Fields:

ADDR unsigned int
Address
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_watchpoint_far

Fields:

LOWER_BOUND unsigned int
the lowest address accessed by the instruction that triggered the watchpoint
UPPER_BOUND unsigned int
the highest watchpointed address accessed by that instruction

ArchMsg.Warning.warning_AdvSIMDExpandImmUnexpectedZero

AdvSIMDExpandImm may treat this immediate value as UNPREDICTABLE.

ArchMsg.Warning.warning_ConditionalSMC

SMC instruction has UNPREDICTABLE effects when conditional when combined with traps.

ArchMsg.Warning.warning_access_crosses_page_boundary

Access crosses page boundary.

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault_on_both_pages

Access crosses page boundary has fault(s).

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_access_crosses_page_boundary_spanning_different_memory

Fields:

ADDR unsigned int
: address of access crossing page

MEMTYPE_PAGE1 signed int
: First Page's memory type (0-Normal, 1-Non-Normal)

MEMTYPE_PAGE2 signed int
: Second Page's memory type (0-Normal, 1-Non-Normal)

ArchMsg.Warning.warning_access_wraps_around_memory

Fields:

WIDTH unsigned int
: Address width

ArchMsg.Warning.warning_bx_from_thumbee

Fields:

ADDR unsigned int
: Destination address

ArchMsg.Warning.warning_ccsidr_unimplemented_level

Fields:

CSSELR unsigned int
: current effective value of CSSELR

ArchMsg.Warning.warning_change_to_ns_when_tge_set

Attempting to change NS to 1 when HCR.TGE = 1.

ArchMsg.Warning.warning_contiguous_bit_check_abort

Fields:

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ArchMsg.Warning.warning_contiguous_bit_error

Fields:

CONTIG_ADDR unsigned int
: address of the TLB first read, with which this entry is expected to be contiguous

CONTIG_DATA unsigned int
: expected contents based on the entry at CONTIG_ADDR

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ENTRY_DATA unsigned int
: contents of conflicting TLB entry

ArchMsg.Warning.warning_cp10_cp11_mismatch

Fields:

EL unsigned int
: EL owning the control

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_cp10_cp11_reserved_value

Fields:

CP10 unsigned int
: CP10 access

CP11 unsigned int
: CP11 access

ArchMsg.Warning.warning_csselr_level_out_of_range

Fields:

IMPLEMENTED unsigned int
: number of cache levels implemented

LEVEL unsigned int
: written value of CSSELR.Level

TYPE enum
: The type of cache selected by CSSELR.InD and TnD

ArchMsg.Warning.warning_debug_flow_control_bits_not_obeyed

Fields:

ERROR enum
: type of error resulting

REGISTER enum
: register under consideration

ArchMsg.Warning.warning_debug_register_access_during_reset

Fields:

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

ArchMsg.Warning.warning_decode_cps_inconsistent_fields

The mode setting fields in the CPS instruction are inconsistent.

ArchMsg.Warning.warning_decode_invalid_state

Instruction is unpredictable in current exception-level/security state.

ArchMsg.Warning.warning_default_cacheable_mmu_on

Default cacheable is enabled (HCR.DC = 1) while MMU is enabled.

ArchMsg.Warning.warning_default_cacheable_vmmu_off

Default cacheable is enabled (HCR.DC = 1) while HCR.VM = 0.

ArchMsg.Warning.warning_deprecated_wvr_bit_2_set

Fields:

N unsigned int
: index

WVR unsigned int
: DBGWVR

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_eret_while_software_step_active_pending

Missing ISB between setting MDSCR_EL1.SS and ERET.

ArchMsg.Warning.warning_exclusive_to_non_normal

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_exclusive_to_non_writeback

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_execute_from_device_memory

An attempt was made to execute from Device memory.

ArchMsg.Warning.warning_illegal_cpsr_mode

Fields:

NEW_MODE unsigned int
: New value of CPSR.M

ArchMsg.Warning.warning_illegal_srs_mode

Fields:

MODE unsigned int
: mode for Banked SP

ArchMsg.Warning.warning_implementation_defined_read_debug_register_in_SCS

Read Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_implementation_defined_sequential_security_transitions_supported

The behaviour of a sequential instruction fetches that cross from non-secure to secure memory and contain SG instruction is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_implementation_defined_stack_limit_check_supported

It is IMPLEMENTATION DEFINED whether stack pointer limit checking is performed for the this instructions. Fields:

ADDRESS unsigned int
: Address of instruction

OPCODE unsigned int
: instruction opcode

ArchMsg.Warning.warning_implementation_defined_write_debug_register_in_SCS

Write Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

DATA unsigned int
: data attempted to be written

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PPB_OFFSET unsigned int
debug register offset

ArchMsg.Warning.warning_invalid_tcr_granule

Fields:

REQUEST signed int
page size requested (or 0 for reserved)
SUBSTITUTE signed int
best guess available page size
TG_ID bool
bits TG0 or TG1

ArchMsg.Warning.warning_load_multiple_user_registers_from_user_mode

An LDM(user registers) instruction executed from user mode.

ArchMsg.Warning.warning_load_pc_from_unaligned

PC loaded from an unaligned location.

ArchMsg.Warning.warning_mte_in_mem_attr_not_supported

Fields:

ATTR unsigned int
Memory attribute

ArchMsg.Warning.warning_reserved_breakpoint_state_match

Fields:

IS_BREAKPOINT unsigned int
Is breakpoint, not watchpoint

ArchMsg.Warning.warning_shareability

Unpredictable: combination of the 1st and 2nd stages of translation is Normal Inner Non-Cacheable, Outer Non-Cacheable.

ArchMsg.Warning.warning_software_step_set_while_enabled

MDSCR_EL1.SS set to 1 while software step debug exceptions are enabled.

ArchMsg.Warning.warning_thumb_instruction_wraps_around_memory

Thumb instruction warps around memory.

ArchMsg.Warning.warning_ttbr_sbz_bits_are_not_zero

Fields:

MSB unsigned int
x-1
TTBR enum
which TTBR register

ArchMsg.Warning.warning_unaligned_address_dbgdtrrx_write

Fields:

ADDR unsigned int
Address

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unaligned_address_dbgdtrtx_read

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_unaligned_to_device

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_unaligned_to_strongly_ordered

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_unknown_sau_rnr

SAU_RNR was set to an unsupported value.

ArchMsg.Warning.warning_unpredictable_AIRCR_PRIP_and_BFHFNMINP

The effect of setting both AIRCR.BFHFNMINP and AIRCR.PRIP to 1 is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTCLRACTIVE_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTCLRACTIVE if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_and_SYSRESETREQ

When the processor is halted in Debug state, if a write to the register writes a 1 to both VECTRESET and SYSRESETREQ, the behavior is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTRESET if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_incorrect_VKEY

The value 0x05FA must be written to AIRCR.VKEY, otherwise the register write is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_EXE_RETURN_Reserved_Bit

EXC_RETURN[23:7] are reserved with the special condition that all bits should be written as one. Values other than all 1s are UNPREDICTABLE. Fields:

EXC_RETURN unsigned int
: Special PC value

SBO_MASK unsigned int
: Which bits Should Be One to not UNPRED

ArchMsg.Warning.warning_unpredictable_change_to_DEMCR_MON_STEP_at_insufficient_priority

The effect of changing DEMCR.MON_STEP at an execution priority that is lower than the priority of the DebugMonitor exception is CONSTRAINED UNPREDICTABLE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_change_to_priority_of_active_exception

Changing the priority of an active exception is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_clear_lspact

Fields:

DRVEXC_TAKEN unsigned int
: derived exception taken not pended

REACHED_LAST_STORE unsigned int
: only the last faulted

ArchMsg.Warning.warning_unpredictable_dbg_exit_unaligned_dlr

Debug exit to AArch32 with DLR[0] set to 1. PC[0] can be set to 0 or to DLR[0].

ArchMsg.Warning.warning_unpredictable_exception_catch

Generation of an exception catch event is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_inconsistent_state

state on exception return is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_instruction

exception return instruction is unpredictable.

ArchMsg.Warning.warning_unpredictable_in_debug_state

Instruction is UNPREDICTABLE when executed in debug state.

ArchMsg.Warning.warning_unpredictable_mair_encoding

Fields:

ATTR unsigned int
: Memory attribute

ArchMsg.Warning.warning_unpredictable_pmu_counter_access

Access to PMU counters from non-secure EL0 or EL1 is UNPREDICTABLE with MDCR_EL2.HPMN set to 0.

ArchMsg.Warning.warning_unpredictable_prioritization_breakpoint_match_vector_catch

Two events with the same priority occurred at the same instruction: Address Matching Vector Catch debug event, and Breakpoint debug event It is constrained unpredictable which is taken.

ArchMsg.Warning.warning_unpredictable_stack_selection

UNPREDICTABLE state on context switch.

ArchMsg.Warning.warning_unpredictable_tsize_out_of_range

Fields:

TCR enum
: Which TCR

TSIZE signed int
: Value of TSize

TSIZE_MAX signed int
: The maximum allowed value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

TTBR unsigned int
: Which TTBR region

ArchMsg.Warning.warning_unpredictable_unaligned_pc_as_base_register

The PC value must be word-aligned, otherwise the behavior of the instruction is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_unaligned_pop_stack

Fields:

FRAME_ADDR unsigned int
: unaligned stack frame address

ArchMsg.Warning.warning_unpredictable_vmsa_memattrib

Fields:

TEXCB unsigned int
: texcb value

TEX_REMAP bool
: use tex remap

ArchMsg.Warning.warning_unpredictable_vtcr_t0sz_sl0

Fields:

SL0 signed int
: VTCR.SL0, starting level for stage 2 translation table walks

T0SZ signed int
: VTCR.T0SZ, required input address range

ArchMsg.Warning.warning_unpredictable_write_DHCSR

Fields:

NEW unsigned int
: data attempted to be written

OLD unsigned int
: previous value

ArchMsg.Warning.warning_unpredictable_write_SHCSR

UNPREDICTABLE setting a pending bit to 1 for an exception with priority >= execution priority.

ArchMsg.Warning.warning_unsupported_access_to_memory_mapped_register

Access to memory mapped register is unsupported.

ArchMsg.Warning.warning_user_jmcr_access

Fields:

WRITE unsigned int
: access is write

ArchMsg.Warning.warning_wcr_mask_and_bas

DBGWCR_EL1.MASK is non-zero and BAS is not 0xff.

ArchMsg.Warning.warning_wcr_mask_reserved

DBGWCR_ELn.MASK is set to a reserved value.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_wcr_non_configuous_bas

DBGWCR_EL1.BAS has non-contiguous set of ones.

ArchMsg.Warning.warning_wvr_masked_bit_not_zero

DBGWVRn_EL1.VA contains a bit masked by DBGWCRn_EL1.MASK that is not zero.

ArchMsg.Why.why_branch_target_exception

A branch target exception has occurred. Fields:

BTYPE unsigned int
: pstate.BTYPE at time branch taken

OPCODE unsigned int
: opcode at branch target

PC unsigned int
: Virtual Address of branch target

ArchMsg.Why.why_illegal_state

An illegal mode change has occured. Fields:

MESSAGE string
: The message

BRANCH_MISPREDICT

Simulating branch mispredict. Fields:

PC unsigned int
: Origin address (or 0 if unavailable)

BRA_DIR

Direct branches, to immediate address. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR

Indirect branches, perhaps to a register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR_CCFAIL

Indirect branches fail, perhaps to a register.

BRA_RET_CCFAIL

Branche return fail, perhaps to a register.

CACHE_MAINTENANCE_OP

Cache Maintenance Operation. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Specified MVA or Set/way

FUNCTION enum
: Clean / Invalidate

SCOPE enum
: Affected region

SIDE enum
: Inst / Data.

CCFAIL

Conditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCFAIL_UNC

Unconditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS

Conditional instruction condition check pass. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS_UNC

Unconditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CHECKPOINT_MESSAGE

Report error messages from the checkpointing process. Fields:

message string
: Message contents.

CHECKPOINT_RESTORE_END

Checkpoint restore completed.

CHECKPOINT_RESTORE_START

Checkpoint restore about to start.

CHECKPOINT_SAVE_END

Checkpoint save completed.

CHECKPOINT_SAVE_START

Checkpoint save about to start.

CODE_CACHE_MAINT

Code cache maintenance. Fields:

END_INCL unsigned int
: Inclusive end addres for by VA or PA requests

NS bool
: Non-secure world for PA

START unsigned int
: Start address for by VA or PA requests

TYPE enum
: Request type.

COMPILE_BLOCK_END

Last instruction of basic block translated. Fields:

VADDR unsigned int
: Address of next instruction after this basic block

COMPILE_BLOCK_START

First instruction of basic block translated. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VADDR unsigned int
: Address of first instruction in basic block

COMPILE_INST

ARM instruction compiled. Fields:

DISASS string
: Disassembly of the instruction.

ISET enum
: The instruction set of this instruction.

ITSTATE unsigned int
: The ITSTATE current for the instruction.

OPCODE unsigned int
: The opcode of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

CONTEXTIDR

Context ID Register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

NS bool
: Secure or nonsecure banked register is accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

CONTEXT_SYNC

Called for every context synchronization event. Fields:

ARCH_CSE bool
: Architectural Context Synchronization Event

SYSREG_SYNC bool
: System register synchronization event

CORE_ENDIAN

Core BE8 Big-Endian state changed. Fields:

BE8 bool
: Core BE8 Big-Endian state

CORE_INFO

Static processor attributes. Only triggered by a call to DumpState(). Fields:

ARCH_PROFILE enum
: The architecture profile of the core.

CLUSTER_ID unsigned int
: The cluster ID of this processor.

CORE_NUM unsigned int
: The number of this core in an MP processor.

DETAILED_CORE_FEATURES enum
: Detailed core features as a bitfield. Enum values are bitmasks to extract the info from the field

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

FPU_VERSION enum
: The VFP version implemented by the core.

MEM_ARCH enum
: The memory architecture of the core.

MIDR_EL1 unsigned int
: MIDR_EL1 value

NUM_CORES unsigned int
: The number of cores in this MP processor.

QUANTUM_SIZE unsigned int
: The default quantum size of the core.

SECURITY_FEATURES bool
: Does the core have security features?

CORE_LOADS

Processor load accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_REGS

Changes of the core registers R0 to R14. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number, 0 to 14.

MODE enum
: Bank of the register accessed.

OLD_VALUE unsigned int
: The old value overwritten.

PHYS_ID enum
: The physical register accessed.

VALUE unsigned int
: The new value written to the register.

CORE_REGS64

Changes of the core registers X0..X30, SP_ELn. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: The register number, 0..30 for X0..X30, >=32 for SP_ELn.

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

CORE_STORES

Processor store accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CP14_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP14_WRITE

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

CP15_WRITE

System Control Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

CPSR

CPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

OLD_VALUE unsigned int
: The old CPSR value

UNKNOWN unsigned int
: Bits within the register that have unknown value.

VALUE unsigned int
: The new CPSR value

CRYPTO_SPEC

Every crypto instruction speculatively executed.

CorePowerStateChange

Core power-up state has changed. Fields:

PU bool
: Whether the core powered up

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DATA_CACHE_SET_ALLOC_TAGS

Set allocation tags from DC GVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of memory in bytes.

TAG unsigned int
: The value of the tag being set.

VADDR unsigned int
: Virtual address used for setting allocation tags.

DATA_CACHE_ZERO

Zero and invalidate cache line from DC ZVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of zero'd memory in bytes.

VADDR unsigned int
: Virtual address of the zero'd cache line.

DEBUG_EVENT

Hardware debug support event. Fields:

EVENT enum
: Description of event

VALUE unsigned int
: data value

END_COMPILE

Compilation end. Fields:

END_OF_PAGE_COUNT unsigned int
: Number of basic blocks exited due to page boundary since START_COMPILE.

FETCHFAIL_COUNT unsigned int
: Number of basic blocks exited due to ifetch failure START_COMPILE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: Number of instructions compiled since START_COMPILE.

NONSEQ_COUNT unsigned int
: Number of basic blocks exited due to non-sequential instructions since START_COMPILE.

PAGE_STRADDLE_COUNT unsigned int
: Number of basic blocks exited due to unaligned instructions crossing page since START_COMPILE.

ESB

Error Synchronization Barrier hit. Fields:

ASYNC bool
: ESB is async

IMPLICIT bool
: ESB is implicit

EXCEPTION

Exception taken. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

IS_PHYSICAL bool
: Physical or Virtual exception

LR unsigned int
: The value assigned to the link register.

PC unsigned int
: The location where the exception occurred.

PREFERRED_RETURN unsigned int
: The preferred return address for the exception.

TARGET_ISET enum
: The instruction set of the exception handler code.

TARGET_PC unsigned int
: The address the exception branches to.

VECTOR enum
: The exception vector.

VECTOR_OFFSET unsigned int
: The offset of an exception vector from the base address.

EXCEPTION_END

Every exception completed.

EXCEPTION_RAISE

Every exception raised.

EXCEPTION_RETURN

Branches on leaving exception. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

EXCEPTION_RETURN_PREBRANCH

Exception return event that must occur before PC and processor state have updated.

EXCEPTION_START

Every exception started.

FIQ_TAKEN

FIQ taken exception.

FP_STATE

Floating point state. Fields:

FPSR unsigned int
: FPSR register value.

MASK unsigned int
: The mask indicating updated cumulative bits.

HLT

HLT instruction occurred. Fields:

IMM16 unsigned int
: The 16-bit immediate value encoded in HLT instruction.

INST

Every instruction executed. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DEBUG_STATE bool
: The instruction is executed in debug state.

DISASS string
: Disassembly of instruction.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The current instruction set.

ITSTATE unsigned int
: The current ITSTATE.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

MODE enum
: The mode the core is in.

NS unsigned int
: The core's non-secure bit.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

OPCODE unsigned int
: The opcode of the instruction.

PADDR unsigned int
: The physical address of the instruction.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

INST_END

Every instruction completed.

INST_START

Every instruction started. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

INST_STRADDLE

Instruction straddles boundary. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

IRQ_TAKEN

IRQ taken exception.

ISB

ISB instruction occurred.

LOAD_MULTIPLE_REGS_LIST

List of destination registers (not source) of a multiple registers load instruction. Fields:

REG string
: Register to add into the list

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOCAL_MONITOR

Local monitor activity. Fields:

PADDR unsigned int
: Local Monitor Address

State enum
: State of the monitor (Open/Exclusive)

MEMMAP_DEBUG_READ

Memory mapped reads to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

MEMMAP_DEBUG_WRITE

Memory mapped writes to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

MMU_TRANS

Address translation information. Fields:

ASID unsigned int
: Address space identifier.

CORE_NUM unsigned int
: Core number in a multi processor.

Hyp bool
: Entry matches in Hyp state only

INNERCACHE_RA bool
: Is the inner cache allocate on read

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INNERCACHE_TRANSIENT bool
: Is the inner write-through transient

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

MEMTYPE enum
: Memory type.

NSDESC enum
: Is secure side supposed to access secure or nonsecure memory

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TRANSIENT bool
: Is the outer write-through transient

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

PADDR unsigned int
: Physical address of the access.

PAGESIZE unsigned int
: Page size as log2(size).

SH enum
: Shareability

SIDE enum
: Inst / Data.

STAGE1_PERM unsigned int
: Stage 1 permissions, [5:3] Privileged access (XWR) [2:0] User access (XWR)

STAGE2_PERM unsigned int
: Stage 2 permission mask [5:3] Privileged access (XWR) [2:0] User access (XWR)

VADDR unsigned int
: Virtual address of the access.

VMID unsigned int
: Virtual machine identifier.

XS bool
: XS Attribute

nG enum
: Flag indicating whether ASID will be matched

MMU_TTB_READ

This event is triggered by reads caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data read.

IPA unsigned int
: For stage 1, the IPA of the read.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OUTADDR unsigned int
: The output physical address.

PADDR unsigned int
: The physical address of the read.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MMU_TTB_WRITE

This event is triggered by writes caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data written.

IPA unsigned int
: For stage 1, the IPA of the write.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

PADDR unsigned int
: The physical address of the write.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MODE_CHANGE

Mode change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: The new mode.

NON_SECURE enum
: The new security state

OLD_MODE enum
: The old mode.

NOT_TAG_CHECKED_INST

MTE Tag not checked for current instruction.

PERIODIC

Called for every quantum. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PC unsigned int
The address of the next instruction to be executed on this CPU.

PMU_COUNTER_OVERFLOW

PMU counter overflow. Fields:

EVENT_ID unsigned int
Event Identifier
INDEX unsigned int
Counter index (as selected by PMSELR).
INTERRUPT bool
Is interrupt enabled on overflow for this counter

PREFETCH_MEMORY64

Prefetch from PRFM or PRFUM instructions. Fields:

PRFOP unsigned int
Prefetch hint
VADDR unsigned int
Virtual address of the location that should be prefetched.

PRELOAD_DATA

Data preload from PLD instruction. Fields:

VADDR unsigned int
Virtual address of the data that should be preloaded.

PRELOAD_INST

Instruction preload from PLI instruction. Fields:

VADDR unsigned int
Virtual address of the instruction that should be preloaded.

PRE_CORE_LOAD

Trace just before a core load. Fields:

CURRENT_TIME unsigned int
The core's current time, as simulated time plus local time.
LOCAL_TIME unsigned int
The core's local time, relative to the current quantum.

PRE_CORE_STORE

Trace just before a core store. Fields:

CURRENT_TIME unsigned int
The core's current time, as simulated time plus local time.
LOCAL_TIME unsigned int
The core's local time, relative to the current quantum.

PRE_TTB_READ

This event is triggered before reads caused by a translation table walk. Fields:

ADDR unsigned int
The physical address of the read.

PSTATE_PACTIVE_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions on PChannel. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

PSTATE_PACTIVE_OPMODE_TRANSITION_DENIED

Trace the denied pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_POWER_TRANSITION

Trace pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

PSTATE_PACTIVE_POWER_TRANSITION_DENIED

Trace the denied pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_THREAD_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions for threads on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

RUN_STATE

Run state transition. Fields:

INST_COUNT unsigned int
: Ticks count at point of transition.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NEW enum
: New run state.

OLD enum
: Old run state.

SEMIHOSTING_CALL

Call of a semihost function occurred. Fields:

PC unsigned int
: The program counter after the semihosting call.

SEMIHOSTING_PRECALL

About to call semihost. Fields:

EL2 bool
: Use EL2 translation regime.

NS bool
: Is Non-Secure.

REG_WIDTH unsigned int
: The current register width in bytes.

SIGNAL

External signal state change. Fields:

SIGNAL enum
: Signal that changed

STATE bool
: Signal asserted state

SOFTWARE_STEP

Return the Debug Software Step state. Fields:

STATE enum
: Software Step state

SPSR

SPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: Which of the banked SPSR registers is written.

OLD_VALUE unsigned int
: The old SPSR value

VALUE unsigned int
: The new SPSR value

START_COMPILE

Compilation started. Fields:

VADDR unsigned int
: Instruction where compilation begins.

STG_MULTIPLE_MICRO_OPS_ADDRESS_RANGE

Address range for MTE stores with multiple uops. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

STORE_MULTIPLE_REGS_ADDRESS_RANGE

Address range for stores with multiple registers. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

SUBOPTIMAL_LDST_RETIRED

suboptimal load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1: half-word, 2: word ...)

OPTIMAL_ALIGN_SELECTOR unsigned int
: Selector to decide the optimal alignment size (1: access width, 2: 4-bytes, 3: 8-bytes, ..., 12: 4-kbytes)

VADDR unsigned int
: The virtual address of the access

SYNC

Called for every synchronization, that is every quantum break. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

LOCAL_QUANTUM unsigned int
: The local quantum of this CPU.

LOCAL_TIME unsigned int
: The local time of this CPU.

SYNC_REASON enum
: Reason for sync.

SYSCALL

System call instruction executed. Fields:

IMM unsigned int
: Immediate value of the system call instruction.

TYPE enum
: System call type.

VADDR unsigned int
: Instruction that caused the system call.

SYSREG_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE32

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE64

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

UNALIGNED_LDST_RETIRED

Processor unaligned load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1=>half-word, 2->word ...)

NUMBER_OF_BEATS unsigned int
: Number of accesses (beats) in this burst (Total size in bytes is ACCESS_SIZE * NUMBER_OF_BEATS)

VADDR unsigned int
: The virtual address of the access

VFP_D_REGS

VFP/NEON D 64 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_Q_REGS

VFP/NEON Q 128 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding byte in the register. For example, 0x1 would indicate a write to VALUE[7:0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_SYS_REGS

Writes to the VFP/NEON units system registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: Which VFP system register is written.

MASK unsigned int
: The mask indicating updated cumulative bits.

OLD_VALUE unsigned int
: The register's old value overwritten.

VALUE unsigned int
: The new value written to the VFP system register.

VFP_S_REGS

VFP/NEON S 32 bit register write. Fields:

ALIASING enum
: Type of register aliasing used.

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

WAYPOINT

Signals end of basic block execution. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: Origin instructions set

IS_COND bool
: Indicates if this is a conditional waypoint.

PC unsigned int
: Origin address (or 0 if unavailable)

TAKEN bool
: Indicates if this waypoint was taken.

TARGET unsigned int
: Destination address

TARGET_ISET enum
: Destination instructions set

WFE_END

WFE ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFE.

WFE_EVENT_REGISTER

WFE event register status: set/clear, reason. Fields:

INST_COUNT unsigned int
: Ticks count.

REASON enum
: Reason for set/clear. Only REASON==0 (Cleared by WFE) clears the bit, all other reasons set it.

WFE_IGNORED

WFE ignored. Fields:

EVENT bool
: This WFE was ignored because the event register was set.

INST_COUNT unsigned int
: Ticks count when ignoring WFE.

TRAPPED bool
: This WFE was trapped.

WFE_START

WFE entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFE.

WFI_END

WFI ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFI.

WFI_IGNORED

WFI ignored. Fields:

DISABLED bool
: This WFI was ignored because WFI is disabled.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: Ticks count when ignoring WFI.

REASON enum
: specifies reason why WFI trace was ignored

TRAPPED bool
: This WFI was trapped.

WFI_START

WFI entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFI.

WFI_WAKEUP

WFI wakeup. Fields:

INST_COUNT unsigned int
: Ticks count when WFI wakeup occurred.

REASON enum
: Reason for wakeup.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.12 ARM_Cortex-A55 - trace

This section describes the trace sources.

ASYNC_MEMORY_FAULT

Context ID Register write. Fields:

FAULT unsigned int
: Fault status in ESR format

PADDR unsigned int
: Physical Address (or 0 if unavailable)

VADDR signed int
: Virtual Address (or 0 if unavailable)

ATOMIC_END_ACCESS

Bus trace access for atomics. Fields:

ACCESS_FAIL bool
: Memory access failed

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

LOAD_VALUE unsigned int
: Loaded values

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ATOMIC_START_ACCESS

Bus trace access for atomics. Fields:

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

NSDESC unsigned int
: The security state of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ArchMsg.Error.exit_code

Fields:

EXITCODE signed int
: Exit code to be returned

KIND string
: Component kind that invoked the exit code trace

REASON string
: Reason why the exit code trace was invoked (optional)

ArchMsg.Warning.IgnoredWarmResetRequest

Event fires when software attempts to request warm reset through write to deprecated DBGPRCR.CWRR.

ArchMsg.Warning.RAS_ErrselrUnpredictableState

ERRSELR is in unpredictable state due to value greater than number_of_error_records written to it.

ArchMsg.Warning.RAS_UnpredictableStateWrite

Fields:

REG_NAME string
: Name of register being written.

ArchMsg.Warning.TarmacReconstructor

Unexpected event happened during Tarmac reconstruction. Fields:

REASON string
: Reason

ArchMsg.Warning.Unpredictably Indexed PM Event Register

Fields:

IsDirect unsigned int
: Direct Access rather than indirect

N unsigned int
: Number of Accessible Registers

SEL unsigned int
: Selector

ArchMsg.Warning.branch_to_unaligned_address

Fields:

ADDR unsigned int
: Unaligned branch address

ArchMsg.Warning.cache_contents_unknown

A cache has been activated without being invalidated and may contain UNKNOWN content. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

REGIME unsigned int
: Current translation regime

SIDE unsigned int
: 0 d-side cache or unified cache, 1 i-side

ArchMsg.Warning.dcimvac_matches_watchpoint

Watchpoints matching an AArch32 DCIMVA are implementation defined.

ArchMsg.Warning.decode_fieldmismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_fracbitsnegative

VCVT Instruction is unpredictable with a negative number of fraction bits.

ArchMsg.Warning.decode_initblock

Instruction is unpredictable in an IT block.

ArchMsg.Warning.decode_initblocknotlast

Instruction is unpredictable in an IT block when not the last.

ArchMsg.Warning.decode_invalid_condition

Instruction is unpredictable when the condition is not AL.

ArchMsg.Warning.decode_invalidfieldcombination

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_invalidvalue

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_mem_hint_unallocated

There is no memory hint allocated to this bit pattern.

ArchMsg.Warning.decode_registermatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_registermismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registeroutofrange

Float point instruction is unpredictable indexing beyond the end of the register bank.

ArchMsg.Warning.decode_sbzsbo

Reserved bits in the instruction are not canonical.

ArchMsg.Warning.decode_transactiontoobig

Attempt to perform a memory transaction of more than 128 bytes.

ArchMsg.Warning.decode_unpred_other

Instruction is unpredictable.

ArchMsg.Warning.decode_unpreduseofpc

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13inreglist

Instruction is unpredictable with R13 in register-list.

ArchMsg.Warning.decode_useofd16orhigher

Attempt to use register D16 or higher where it is not allowed.

ArchMsg.Warning.decode_writebackandbaseinlist

Instruction is unpredictable with write back when the base register is in the transfer list.

ArchMsg.Warning.decode_zeroregistersinlist

Instruction is unpredictable when no registers are to be transfered.

ArchMsg.Warning.recursive_branch

An instruction is performing a branch that targets the same instruction. If this was intended then a WFI might be more effective. Fields:

PC unsigned int
: Address of instruction causing the branch

ArchMsg.Warning.recursive_exception

An instruction has generated an exception that targets the same instruction. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CPSR unsigned int
: Processor state

PC unsigned int
: Address of instruction causing the exception

TYPE enum
: Type of exception

ArchMsg.Warning.reentrant_vector_catch

Fields:

ADDR unsigned int
: vector address

ArchMsg.Warning.reserved_it_state

Fields:

ITSTATE unsigned int
: reserved IT state

ArchMsg.Warning.tlb_contents_unknown

Translation has been enabled without invalidating the TLB. Fields:

EL unsigned int
: Exception level of translation regime

INVALIDITY unsigned int
: Which TLB may be invalid (if TLBs are separate)

NON_SECURE unsigned int
: Security state of translation regime

VMID unsigned int
: Current VMID, if applicable

ArchMsg.Warning.unknown_DLR_DSPSR

Exiting debug state with unknown DLR or DSPSR.

ArchMsg.Warning.unknown_ELR_SPSR

Returning from debug exception with unknown ELR or SPSR.

ArchMsg.Warning.unpredictable_a32_breakpoint

Fields:

ADDR1 unsigned int
: Address

ADDR2 unsigned int
: Breakpoint address

BAS unsigned int
: BAS Field

IS_ADDRESS_MISMATCH_BREAKPOINT bool
: Is Mismatch breakpoint, not a match breakpoint

N unsigned int
: Breakpoint number

OFFSET unsigned int
: Indicates which halfword actually considering into the A32 instruction

ArchMsg.Warning.unpredictable_banked_register_access

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

R unsigned int
R bit
SYSM unsigned int
SYSm

ArchMsg.Warning.unpredictable_ccfail_undef

Fields:

OPCODE unsigned int
opcode of the instruction
VADDR unsigned int
Virtual address of the instruction being executed

ArchMsg.Warning.unpredictable_ici_bits

ICI bits do not apply or are architecturally unpredictable for instruction.

ArchMsg.Warning.unpredictable_t32_breakpoint

Fields:

ADDR1 unsigned int
Address
ADDR2 unsigned int
Breakpoint address
BAS unsigned int
BAS Field
IS_ADDRESS_MISMATCH_BREAKPOINT bool
Is Mismatch breakpoint, not a match breakpoint
N unsigned int
Breakpoint number
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_t32_vector_catch

Fields:

ADDR unsigned int
Address
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_watchpoint_far

Fields:

LOWER_BOUND unsigned int
the lowest address accessed by the instruction that triggered the watchpoint
UPPER_BOUND unsigned int
the highest watchpointed address accessed by that instruction

ArchMsg.Warning.warning_AdvSIMDExpandImmUnexpectedZero

AdvSIMDExpandImm may treat this immediate value as UNPREDICTABLE.

ArchMsg.Warning.warning_ConditionalSMC

SMC instruction has UNPREDICTABLE effects when conditional when combined with traps.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_access_crosses_page_boundary

Access crosses page boundary.

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault_on_both_pages

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_spanning_different_memory

Fields:

ADDR unsigned int
: address of access crossing page

MEMTYPE_PAGE1 signed int
: First Page's memory type (0-Normal, 1-Non-Normal)

MEMTYPE_PAGE2 signed int
: Second Page's memory type (0-Normal, 1-Non-Normal)

ArchMsg.Warning.warning_access_wraps_around_memory

Fields:

WIDTH unsigned int
: Address width

ArchMsg.Warning.warning_bx_from_thumbee

Fields:

ADDR unsigned int
: Destination address

ArchMsg.Warning.warning_ccsidr_unimplemented_level

Fields:

CSSELR unsigned int
: current effective value of CSSELR

ArchMsg.Warning.warning_change_to_ns_when_tge_set

Attempting to change NS to 1 when HCR.TGE = 1.

ArchMsg.Warning.warning_contiguous_bit_check_abort

Fields:

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ArchMsg.Warning.warning_contiguous_bit_error

Fields:

CONTIG_ADDR unsigned int
: address of the TLB first read, with which this entry is expected to be contiguous

CONTIG_DATA unsigned int
: expected contents based on the entry at CONTIG_ADDR

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ENTRY_DATA unsigned int
: contents of conflicting TLB entry

ArchMsg.Warning.warning_cp10_cp11_mismatch

Fields:

EL unsigned int
: EL owning the control

ArchMsg.Warning.warning_cp10_cp11_reserved_value

Fields:

CP10 unsigned int
: CP10 access

CP11 unsigned int
: CP11 access

ArchMsg.Warning.warning_csselr_level_out_of_range

Fields:

IMPLEMENTED unsigned int
: number of cache levels implemented

LEVEL unsigned int
: written value of CSSELR.Level

TYPE enum
: The type of cache selected by CSSELR.InD and TnD

ArchMsg.Warning.warning_debug_flow_control_bits_not_obeyed

Fields:

ERROR enum
: type of error resulting

REGISTER enum
: register under consideration

ArchMsg.Warning.warning_debug_register_access_during_reset

Fields:

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

ArchMsg.Warning.warning_decode_cps_inconsistent_fields

The mode setting fields in the CPS instruction are inconsistent.

ArchMsg.Warning.warning_decode_invalid_state

Instruction is unpredictable in current exception-level/security state.

ArchMsg.Warning.warning_default_cacheable_mmu_on

Default cacheable is enabled (HCR.DC = 1) while MMU is enabled.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_default_cacheable_vmmu_off

Default cacheable is enabled (HCR.DC = 1) while HCR.VM = 0.

ArchMsg.Warning.warning_deprecated_wvr_bit_2_set

Fields:

N unsigned int
: index

WVR unsigned int
: DBGWVR

ArchMsg.Warning.warning_eret_while_software_step_active_pending

Missing ISB between setting MDSCR_EL1.SS and ERET.

ArchMsg.Warning.warning_exclusive_to_non_normal

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_exclusive_to_non_writeback

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_execute_from_device_memory

An attempt was made to execute from Device memory.

ArchMsg.Warning.warning_illegal_cpsr_mode

Fields:

NEW_MODE unsigned int
: New value of CPSR.M

ArchMsg.Warning.warning_illegal_srs_mode

Fields:

MODE unsigned int
: mode for Banked SP

ArchMsg.Warning.warning_implementation_defined_read_debug_register_in_SCS

Read Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_implementation_defined_sequential_security_transitions_supported

The behaviour of a sequential instruction fetches that cross from non-secure to secure memory and contain SG instruction is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_implementation_defined_stack_limit_check_supported

It is IMPLEMENTATION DEFINED whether stack pointer limit checking is performed for the this instructions. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDRESS unsigned int
: Address of instruction

OPCODE unsigned int
: instruction opcode

ArchMsg.Warning.warning_implementation_defined_write_debug_register_in_SCS

Write Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

DATA unsigned int
: data attempted to be written

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_invalid_tcr_granule

Fields:

REQUEST signed int
: page size requested (or 0 for reserved)

SUBSTITUTE signed int
: best guess available page size

TG_ID bool
: bits TG0 or TG1

ArchMsg.Warning.warning_load_multiple_user_registers_from_user_mode

An LDM(user registers) instruction executed from user mode.

ArchMsg.Warning.warning_load_pc_from_unaligned

PC loaded from an unaligned location.

ArchMsg.Warning.warning_mte_in_mem_attr_not_supported

Fields:

ATTR unsigned int
: Memory attribute

ArchMsg.Warning.warning_reserved_breakpoint_state_match

Fields:

IS_BREAKPOINT unsigned int
: Is breakpoint, not watchpoint

ArchMsg.Warning.warning_shareability

Unpredictable: combination of the 1st and 2nd stages of translation is Normal Inner Non-Cacheable, Outer Non-Cacheable.

ArchMsg.Warning.warning_software_step_set_while_enabled

MDSCR_EL1.SS set to 1 while software step debug exceptions are enabled.

ArchMsg.Warning.warning_thumb_instruction_wraps_around_memory

Thumb instruction warps around memory.

ArchMsg.Warning.warning_ttbr_sbz_bits_are_not_zero

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MSB unsigned int
x-1
TTBR enum
which TTBR register

ArchMsg.Warning.warning_unaligned_address_dbgdtrrx_write

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_address_dbgdtrtx_read

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_device

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_strongly_ordered

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unknown_sau_rnr

SAU_RNR was set to an unsupported value.

ArchMsg.Warning.warning_unpredictable_AIRCR_PRIP_and_BFHFNMINP

The effect of setting both AIRCR.BFHFNMINP and AIRCR.PRIP to 1 is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTCLRACTIVE_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTCLRACTIVE if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_and_SYSRESETREQ

When the processor is halted in Debug state, if a write to the register writes a 1 to both VECTRESET and SYSRESETREQ, the behavior is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTRESET if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_incorrect_VKEY

The value 0x05FA must be written to AIRCR.VKEY, otherwise the register write is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_EXE_RETURN_Reserved_Bit

EXC_RETURN[23:7] are reserved with the special condition that all bits should be written as one. Values other than all 1s are UNPREDICTABLE. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

EXC_RETURN unsigned int
: Special PC value

SBO_MASK unsigned int
: Which bits Should Be One to not UNPRED

ArchMsg.Warning.warning_unpredictable_change_to_DEMCR_MON_STEP_at_insufficient_priority

The effect of changing DEMCR.MON_STEP at an execution priority that is lower than the priority of the DebugMonitor exception is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_change_to_priority_of_active_exception

Changing the priority of an active exception is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_clear_lspact

Fields:

DRVEXC_TAKEN unsigned int
: derived exception taken not pended

REACHED_LAST_STORE unsigned int
: only the last faulted

ArchMsg.Warning.warning_unpredictable_dbg_exit_unaligned_dlr

Debug exit to AArch32 with DLR[0] set to 1. PC[0] can be set to 0 or to DLR[0].

ArchMsg.Warning.warning_unpredictable_exception_catch

Generation of an exception catch event is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_inconsistent_state

state on exception return is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_instruction

exception return instruction is unpredictable.

ArchMsg.Warning.warning_unpredictable_in_debug_state

Instruction is UNPREDICTABLE when executed in debug state.

ArchMsg.Warning.warning_unpredictable_mair_encoding

Fields:

ATTR unsigned int
: Memory attribute

ArchMsg.Warning.warning_unpredictable_pmu_counter_access

Access to PMU counters from non-secure EL0 or EL1 is UNPREDICTABLE with MDCR_EL2.HPMN set to 0.

ArchMsg.Warning.warning_unpredictable_prioritization_breakpoint_match_vector_catch

Two events with the same priority occurred at the same instruction: Address Matching Vector Catch debug event, and Breakpoint debug event It is constrained unpredictable which is taken.

ArchMsg.Warning.warning_unpredictable_stack_selection

UNPREDICTABLE state on context switch.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_tsize_out_of_range

Fields:

TCR enum
: Which TCR

TSIZE signed int
: Value of TSize

TSIZE_MAX signed int
: The maximum allowed value

TTBR unsigned int
: Which TTBR region

ArchMsg.Warning.warning_unpredictable_unaligned_pc_as_base_register

The PC value must be word-aligned, otherwise the behavior of the instruction is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_unaligned_pop_stack

Fields:

FRAME_ADDR unsigned int
: unaligned stack frame address

ArchMsg.Warning.warning_unpredictable_vmsa_memattrib

Fields:

TEXCB unsigned int
: texcb value

TEX_REMAP bool
: use tex remap

ArchMsg.Warning.warning_unpredictable_vtcr_t0sz_sl0

Fields:

SL0 signed int
: VTCR.SL0, starting level for stage 2 translation table walks

T0SZ signed int
: VTCR.T0SZ, required input address range

ArchMsg.Warning.warning_unpredictable_write_DHCSR

Fields:

NEW unsigned int
: data attempted to be written

OLD unsigned int
: previous value

ArchMsg.Warning.warning_unpredictable_write_SHCSR

UNPREDICTABLE setting a pending bit to 1 for an exception with priority >= execution priority.

ArchMsg.Warning.warning_unsupported_access_to_memory_mapped_register

Access to memory mapped register is unsupported.

ArchMsg.Warning.warning_user_jmcr_access

Fields:

WRITE unsigned int
: access is write

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_wcr_mask_and_bas

DBGWCR_EL1.MASK is non-zero and BAS is not 0xff.

ArchMsg.Warning.warning_wcr_mask_reserved

DBGWCR_ELn.MASK is set to a reserved value.

ArchMsg.Warning.warning_wcr_non_configuous_bas

DBGWCR_EL1.BAS has non-contiguous set of ones.

ArchMsg.Warning.warning_wvr_masked_bit_not_zero

DBGWVRn_EL1.VA contains a bit masked by DBGWCRn_EL1.MASK that is not zero.

ArchMsg.Why.why_branch_target_exception

A branch target exception has occurred. Fields:

BTYPE unsigned int
: pstate.BTYPE at time branch taken

OPCODE unsigned int
: opcode at branch target

PC unsigned int
: Virtual Address of branch target

ArchMsg.Why.why_illegal_state

An illegal mode change has occured. Fields:

MESSAGE string
: The message

BRANCH_MISPREDICT

Simulating branch mispredict. Fields:

PC unsigned int
: Origin address (or 0 if unavailable)

BRA_DIR

Direct branches, to immediate address. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

BRA_INDIR

Indirect branches, perhaps to a register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR_CCFAIL

Indirect branches fail, perhaps to a register.

BRA_RET_CCFAIL

Branche return fail, perhaps to a register.

CACHE_MAINTENANCE_OP

Cache Maintenance Operation. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Specified MVA or Set/way

FUNCTION enum
: Clean / Invalidate

SCOPE enum
: Affected region

SIDE enum
: Inst / Data.

CCFAIL

Conditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCFAIL_UNC

Unconditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS

Conditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS_UNC

Unconditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CHECKPOINT_MESSAGE

Report error messages from the checkpointing process. Fields:

message string
: Message contents.

CHECKPOINT_RESTORE_END

Checkpoint restore completed.

CHECKPOINT_RESTORE_START

Checkpoint restore about to start.

CHECKPOINT_SAVE_END

Checkpoint save completed.

CHECKPOINT_SAVE_START

Checkpoint save about to start.

CODE_CACHE_MAINT

Code cache maintenance. Fields:

END_INCL unsigned int
: Inclusive end addres for by VA or PA requests

NS bool
: Non-secure world for PA

START unsigned int
: Start address for by VA or PA requests

TYPE enum
: Request type.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

COMPILE_BLOCK_END

Last instruction of basic block translated. Fields:

VADDR unsigned int
: Address of next instruction after this basic block

COMPILE_BLOCK_START

First instruction of basic block translated. Fields:

VADDR unsigned int
: Address of first instruction in basic block

COMPILE_INST

ARM instruction compiled. Fields:

DISASS string
: Disassembly of the instruction.

ISET enum
: The instruction set of this instruction.

ITSTATE unsigned int
: The ITSTATE current for the instruction.

OPCODE unsigned int
: The opcode of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

CONTEXTIDR

Context ID Register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

NS bool
: Secure or nonsecure banked register is accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

CONTEXT_SYNC

Called for every context synchronization event. Fields:

ARCH_CSE bool
: Architectural Context Synchronization Event

SYSREG_SYNC bool
: System register synchronization event

CORE_ENDIAN

Core BE8 Big-Endian state changed. Fields:

BE8 bool
: Core BE8 Big-Endian state

CORE_INFO

Static processor attributes. Only triggered by a call to DumpState(). Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ARCH_PROFILE enum
: The architecture profile of the core.

CLUSTER_ID unsigned int
: The cluster ID of this processor.

CORE_NUM unsigned int
: The number of this core in an MP processor.

DETAILED_CORE_FEATURES enum
: Detailed core features as a bitfield. Enum values are bitmasks to extract the info from the field

FPU_VERSION enum
: The VFP version implemented by the core.

MEM_ARCH enum
: The memory architecture of the core.

MIDR_EL1 unsigned int
: MIDR_EL1 value

NUM_CORES unsigned int
: The number of cores in this MP processor.

QUANTUM_SIZE unsigned int
: The default quantum size of the core.

SECURITY_FEATURES bool
: Does the core have security features?

CORE_LOADS

Processor load accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CORE_REGS

Changes of the core registers R0 to R14. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number, 0 to 14.

MODE enum
: Bank of the register accessed.

OLD_VALUE unsigned int
: The old value overwritten.

PHYS_ID enum
: The physical register accessed.

VALUE unsigned int
: The new value written to the register.

CORE_REGS64

Changes of the core registers X0..X30, SP_ELn. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: The register number, 0..30 for X0..X30, >=32 for SP_ELn.

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

CORE_STORES

Processor store accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CP14_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP14_WRITE

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CP15_WRITE

System Control Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

CPSR

CPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

OLD_VALUE unsigned int
: The old CPSR value

UNKNOWN unsigned int
: Bits within the register that have unknown value.

VALUE unsigned int
: The new CPSR value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CRYPTO_SPEC

Every crypto instruction speculatively executed.

CorePowerStateChange

Core power-up state has changed. Fields:

PU bool
: Whether the core powered up

DATA_CACHE_SET_ALLOC_TAGS

Set allocation tags from DC GVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of memory in bytes.

TAG unsigned int
: The value of the tag being set.

VADDR unsigned int
: Virtual address used for setting allocation tags.

DATA_CACHE_ZERO

Zero and invalidate cache line from DC ZVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of zero'd memory in bytes.

VADDR unsigned int
: Virtual address of the zero'd cache line.

DEBUG_EVENT

Hardware debug support event. Fields:

EVENT enum
: Description of event

VALUE unsigned int
: data value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

END_COMPILE

Compilation end. Fields:

END_OF_PAGE_COUNT unsigned int
: Number of basic blocks exited due to page boundary since START_COMPILE.

FETCHFAIL_COUNT unsigned int
: Number of basic blocks exited due to ifetch failure START_COMPILE.

INST_COUNT unsigned int
: Number of instructions compiled since START_COMPILE.

NONSEQ_COUNT unsigned int
: Number of basic blocks exited due to non-sequential instructions since START_COMPILE.

PAGE_STRADDLE_COUNT unsigned int
: Number of basic blocks exited due to unaligned instructions crossing page since START_COMPILE.

ESB

Error Synchronization Barrier hit. Fields:

ASYNC bool
: ESB is async

IMPLICIT bool
: ESB is implicit

EXCEPTION

Exception taken. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

IS_PHYSICAL bool
: Physical or Virtual exception

LR unsigned int
: The value assigned to the link register.

PC unsigned int
: The location where the exception occurred.

PREFERRED_RETURN unsigned int
: The preferred return address for the exception.

TARGET_ISET enum
: The instruction set of the exception handler code.

TARGET_PC unsigned int
: The address the exception branches to.

VECTOR enum
: The exception vector.

VECTOR_OFFSET unsigned int
: The offset of an exception vector from the base address.

EXCEPTION_END

Every exception completed.

EXCEPTION_RAISE

Every exception raised.

EXCEPTION_RETURN

Branches on leaving exception. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

EXCEPTION_RETURN_PREBRANCH

Exception return event that must occur before PC and processor state have updated.

EXCEPTION_START

Every exception started.

FIQ_TAKEN

FIQ taken exception.

FP_STATE

Floating point state. Fields:

FPSR unsigned int
: FPSR register value.

MASK unsigned int
: The mask indicating updated cumulative bits.

HLT

HLT instruction occurred. Fields:

IMM16 unsigned int
: The 16-bit immediate value encoded in HLT instruction.

INST

Every instruction executed. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DEBUG_STATE bool
: The instruction is executed in debug state.

DISASS string
: Disassembly of instruction.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The current instruction set.

ITSTATE unsigned int
: The current ITSTATE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

MODE enum
: The mode the core is in.

NS unsigned int
: The core's non-secure bit.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

OPCODE unsigned int
: The opcode of the instruction.

PADDR unsigned int
: The physical address of the instruction.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

INST_END

Every instruction completed.

INST_START

Every instruction started. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

INST_STRADDLE

Instruction straddles boundary. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

IRQ_TAKEN

IRQ taken exception.

ISB

ISB instruction occurred.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOAD_MULTIPLE_REGS_LIST

List of destination registers (not source) of a multiple registers load instruction. Fields:

REG string
: Register to add into the list

LOCAL_MONITOR

Local monitor activity. Fields:

PADDR unsigned int
: Local Monitor Address

State enum
: State of the monitor (Open/Exclusive)

MEMMAP_DEBUG_READ

Memory mapped reads to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

MEMMAP_DEBUG_WRITE

Memory mapped writes to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

MMU_TRANS

Address translation information. Fields:

ASID unsigned int
: Address space identifier.

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Hyp bool
: Entry matches in Hyp state only

INNERCACHE_RA bool
: Is the inner cache allocate on read

INNERCACHE_TRANSIENT bool
: Is the inner write-through transient

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

MEMTYPE enum
: Memory type.

NSDESC enum
: Is secure side supposed to access secure or nonsecure memory

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TRANSIENT bool
: Is the outer write-through transient

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

PADDR unsigned int
: Physical address of the access.

PAGESIZE unsigned int
: Page size as log2(size).

SH enum
: Shareability

SIDE enum
: Inst / Data.

STAGE1_PERM unsigned int
: Stage 1 permissions, [5:3] Privileged access (XWR) [2:0] User access (XWR)

STAGE2_PERM unsigned int
: Stage 2 permission mask [5:3] Privileged access (XWR) [2:0] User access (XWR)

VADDR unsigned int
: Virtual address of the access.

VMID unsigned int
: Virtual machine identifier.

XS bool
: XS Attribute

nG enum
: Flag indicating whether ASID will be matched

MMU_TTB_READ

This event is triggered by reads caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data read.

IPA unsigned int
: For stage 1, the IPA of the read.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

OUTADDR unsigned int
: The output physical address.

PADDR unsigned int
: The physical address of the read.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MMU_TTB_WRITE

This event is triggered by writes caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data written.

IPA unsigned int
: For stage 1, the IPA of the write.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

PADDR unsigned int
: The physical address of the write.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MODE_CHANGE

Mode change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: The new mode.

NON_SECURE enum
: The new security state

OLD_MODE enum
: The old mode.

NOT_TAG_CHECKED_INST

MTE Tag not checked for current instruction.

PERIODIC

Called for every quantum. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

PC unsigned int
: The address of the next instruction to be executed on this CPU.

PMU_COUNTER_OVERFLOW

PMU counter overflow. Fields:

EVENT_ID unsigned int
: Event Identifier

INDEX unsigned int
: Counter index (as selected by PMSELR).

INTERRUPT bool
: Is interrupt enabled on overflow for this counter

PREFETCH_MEMORY64

Prefetch from PRFM or PRFUM instructions. Fields:

PRFOP unsigned int
: Prefetch hint

VADDR unsigned int
: Virtual address of the location that should be prefetched.

PRELOAD_DATA

Data preload from PLD instruction. Fields:

VADDR unsigned int
: Virtual address of the data that should be preloaded.

PRELOAD_INST

Instruction preload from PLI instruction. Fields:

VADDR unsigned int
: Virtual address of the instruction that should be preloaded.

PRE_CORE_LOAD

Trace just before a core load. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_CORE_STORE

Trace just before a core store. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_TTB_READ

This event is triggered before reads caused by a translation table walk. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR unsigned int
The physical address of the read.

PSTATE_PACTIVE_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Operating mode after the transition
OLD_STATE enum
Operating mode before the transition

PSTATE_PACTIVE_OPMODE_TRANSITION_DENIED

Trace the denied pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Operating mode after the transition
OLD_STATE enum
Operating mode before the transition
REASON string
Reason for the transition being denied

PSTATE_PACTIVE_POWER_TRANSITION

Trace pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Power mode after the transition
OLD_STATE enum
Power mode before the transition

PSTATE_PACTIVE_POWER_TRANSITION_DENIED

Trace the denied pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Power mode after the transition
OLD_STATE enum
Power mode before the transition
REASON string
Reason for the transition being denied

PSTATE_PACTIVE_THREAD_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions for threads on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Operating mode after the transition
OLD_STATE enum
Operating mode before the transition

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

RUN_STATE

Run state transition. Fields:

INST_COUNT unsigned int
: Ticks count at point of transition.

NEW enum
: New run state.

OLD enum
: Old run state.

SEMIHOSTING_CALL

Call of a semihost function occurred. Fields:

PC unsigned int
: The program counter after the semihosting call.

SEMIHOSTING_PRECALL

About to call semihost. Fields:

EL2 bool
: Use EL2 translation regime.

NS bool
: Is Non-Secure.

REG_WIDTH unsigned int
: The current register width in bytes.

SIGNAL

External signal state change. Fields:

SIGNAL enum
: Signal that changed

STATE bool
: Signal asserted state

SOFTWARE_STEP

Return the Debug Software Step state. Fields:

STATE enum
: Software Step state

SPSR

SPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: Which of the banked SPSR registers is written.

OLD_VALUE unsigned int
: The old SPSR value

VALUE unsigned int
: The new SPSR value

START_COMPILE

Compilation started. Fields:

VADDR unsigned int
: Instruction where compilation begins.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

STG_MULTIPLE_MICRO_OPS_ADDRESS_RANGE

Address range for MTE stores with multiple uops. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

STORE_MULTIPLE_REGS_ADDRESS_RANGE

Address range for stores with multiple registers. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

SUBOPTIMAL_LDST_RETIRED

suboptimal load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1: half-word, 2: word ...)

OPTIMAL_ALIGN_SELECTOR unsigned int
: Selector to decide the optimal alignment size (1: access width, 2: 4-bytes, 3: 8-bytes, ..., 12: 4-kbytes)

VADDR unsigned int
: The virtual address of the access

SYNC

Called for every synchronization, that is every quantum break. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

LOCAL_QUANTUM unsigned int
: The local quantum of this CPU.

LOCAL_TIME unsigned int
: The local time of this CPU.

SYNC_REASON enum
: Reason for sync.

SYSCALL

System call instruction executed. Fields:

IMM unsigned int
: Immediate value of the system call instruction.

TYPE enum
: System call type.

VADDR unsigned int
: Instruction that caused the system call.

SYSREG_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE32

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE64

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

UNALIGNED_LDST_RETIRED

Processor unaligned load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1=>half-word, 2->word ...)

NUMBER_OF_BEATS unsigned int
: Number of accesses (beats) in this burst (Total size in bytes is ACCESS_SIZE * NUMBER_OF_BEATS)

VADDR unsigned int
: The virtual address of the access

VFP_D_REGS

VFP/NEON D 64 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ID unsigned int
The register number.
MASK unsigned int
Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]
OLD_VALUE unsigned int
The old value overwritten.
VALUE unsigned int
The new value written to the register.

VFP_Q_REGS

VFP/NEON Q 128 bit register write. Fields:

ALIASING enum
Type of register aliasing used
CORE_NUM unsigned int
Core number in a multi processor.
ID unsigned int
The register number.
MASK unsigned int
Each bit indicates a write to the corresponding byte in the register. For example, 0x1 would indicate a write to VALUE[7:0]
OLD_VALUE unsigned int
The old value overwritten.
VALUE unsigned int
The new value written to the register.

VFP_SYS_REGS

Writes to the VFP/NEON units system registers. Fields:

CORE_NUM unsigned int
Core number in a multi processor.
ID enum
Which VFP system register is written.
MASK unsigned int
The mask indicating updated cumulative bits.
OLD_VALUE unsigned int
The register's old value overwritten.
VALUE unsigned int
The new value written to the VFP system register.

VFP_S_REGS

VFP/NEON S 32 bit register write. Fields:

ALIASING enum
Type of register aliasing used.
CORE_NUM unsigned int
Core number in a multi processor.
ID unsigned int
The register number.
MASK unsigned int
Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]
OLD_VALUE unsigned int
The old value overwritten.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VALUE unsigned int
: The new value written to the register.

WAYPOINT

Signals end of basic block execution. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: Origin instructions set

IS_COND bool
: Indicates if this is a conditional waypoint.

PC unsigned int
: Origin address (or 0 if unavailable)

TAKEN bool
: Indicates if this waypoint was taken.

TARGET unsigned int
: Destination address

TARGET_ISET enum
: Destination instructions set

WFE_END

WFE ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFE.

WFE_EVENT_REGISTER

WFE event register status: set/clear, reason. Fields:

INST_COUNT unsigned int
: Ticks count.

REASON enum
: Reason for set/clear. Only REASON==0 (Cleared by WFE) clears the bit, all other reasons set it.

WFE_IGNORED

WFE ignored. Fields:

EVENT bool
: This WFE was ignored because the event register was set.

INST_COUNT unsigned int
: Ticks count when ignoring WFE.

TRAPPED bool
: This WFE was trapped.

WFE_START

WFE entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFE.

WFI_END

WFI ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFI.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

WFI_IGNORED

WFI ignored. Fields:

DISABLED `bool`
: This WFI was ignored because WFI is disabled.

INST_COUNT `unsigned int`
: Ticks count when ignoring WFI.

REASON `enum`
: specifies reason why WFI trace was ignored

TRAPPED `bool`
: This WFI was trapped.

WFI_START

WFI entered. Fields:

INST_COUNT `unsigned int`
: Ticks count when entering WFI.

WFI_WAKEUP

WFI wakeup. Fields:

INST_COUNT `unsigned int`
: Ticks count when WFI wakeup occurred.

REASON `enum`
: Reason for wakeup.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.13 ARM_Cortex-A57 - trace

This section describes the trace sources.

ASYNC_MEMORY_FAULT

Context ID Register write. Fields:

FAULT unsigned int
: Fault status in ESR format

PADDR unsigned int
: Physical Address (or 0 if unavailable)

VADDR signed int
: Virtual Address (or 0 if unavailable)

ATOMIC_END_ACCESS

Bus trace access for atomics. Fields:

ACCESS_FAIL bool
: Memory access failed

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

LOAD_VALUE unsigned int
: Loaded values

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ATOMIC_START_ACCESS

Bus trace access for atomics. Fields:

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

NSDESC unsigned int
: The security state of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ArchMsg.Error.exit_code

Fields:

EXITCODE signed int
: Exit code to be returned

KIND string
: Component kind that invoked the exit code trace

REASON string
: Reason why the exit code trace was invoked (optional)

ArchMsg.Warning.IgnoredWarmResetRequest

Event fires when software attempts to request warm reset through write to deprecated DBGPRCR.CWRR.

ArchMsg.Warning.TarmacReconstructor

Unexpected event happened during Tarmac reconstruction. Fields:

REASON string
: Reason

ArchMsg.Warning.Unpredictably Indexed PM Event Register

Fields:

IsDirect unsigned int
: Direct Access rather than indirect

N unsigned int
: Number of Accessible Registers

SEL unsigned int
: Selector

ArchMsg.Warning.branch_to_unaligned_address

Fields:

ADDR unsigned int
: Unaligned branch address

ArchMsg.Warning.cache_contents_unknown

A cache has been activated without being invalidated and may contain UNKNOWN content. Fields:

REGIME unsigned int
: Current translation regime

SIDE unsigned int
: 0 d-side cache or unified cache, 1 i-side

ArchMsg.Warning.dcimvac_matches_watchpoint

Watchpoints matching an AArch32 DCIMVA are implementation defined.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_fieldmismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_fracbitsnegative

VCVT Instruction is unpredictable with a negative number of fraction bits.

ArchMsg.Warning.decode_initblock

Instruction is unpredictable in an IT block.

ArchMsg.Warning.decode_initblocknotlast

Instruction is unpredictable in an IT block when not the last.

ArchMsg.Warning.decode_invalid_condition

Instruction is unpredictable when the condition is not AL.

ArchMsg.Warning.decode_invalidfieldcombination

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_invalidvalue

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_mem_hint_unallocated

There is no memory hint allocated to this bit pattern.

ArchMsg.Warning.decode_registermatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registermismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registeroutofrange

Float point instruction is unpredictable indexing beyond the end of the register bank.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_sbzsbo

Reserved bits in the instruction are not canonical.

ArchMsg.Warning.decode_transactiontoobig

Attempt to perform a memory transaction of more than 128 bytes.

ArchMsg.Warning.decode_unpred_other

Instruction is unpredictable.

ArchMsg.Warning.decode_unpreduseofpc

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13inreglist

Instruction is unpredictable with R13 in register-list.

ArchMsg.Warning.decode_useofd16orhigher

Attempt to use register D16 or higher where it is not allowed.

ArchMsg.Warning.decode_writebackandbaseinlist

Instruction is unpredictable with write back when the base register is in the transfer list.

ArchMsg.Warning.decode_zeroregistersinlist

Instruction is unpredictable when no registers are to be transfered.

ArchMsg.Warning.recursive_branch

An instruction is performing a branch that targets the same instruction. If this was intended then a WFI might be more effective. Fields:

PC unsigned int
: Address of instruction causing the branch

ArchMsg.Warning.recursive_exception

An instruction has generated an exception that targets the same instruction. Fields:

CPSR unsigned int
: Processor state

PC unsigned int
: Address of instruction causing the exception

TYPE enum
: Type of exception

ArchMsg.Warning.reentrant_vector_catch

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR unsigned int
vector address

ArchMsg.Warning.reserved_it_state

Fields:

ITSTATE unsigned int
reserved IT state

ArchMsg.Warning.tlb_contents_unknown

Translation has been enabled without invalidating the TLB. Fields:

EL unsigned int
Exception level of translation regime
INVALIDITY unsigned int
Which TLB may be invalid (if TLBs are separate)
NON_SECURE unsigned int
Security state of translation regime
VMID unsigned int
Current VMID, if applicable

ArchMsg.Warning.unknown_DLR_DSPSR

Exiting debug state with unknown DLR or DSPSR.

ArchMsg.Warning.unknown_ELR_SPSR

Returning from debug exception with unknown ELR or SPSR.

ArchMsg.Warning.unpredictable_a32_breakpoint

Fields:

ADDR1 unsigned int
Address
ADDR2 unsigned int
Breakpoint address
BAS unsigned int
BAS Field
IS_ADDRESS_MISMATCH_BREAKPOINT bool
Is Mismatch breakpoint, not a match breakpoint
N unsigned int
Breakpoint number
OFFSET unsigned int
Indicates which halfword actually considering into the A32 instruction

ArchMsg.Warning.unpredictable_banked_register_access

Fields:

R unsigned int
R bit
SYSM unsigned int
SYSm

ArchMsg.Warning.unpredictable_ccfail_undef

Fields:

OPCODE unsigned int
opcode of the instruction

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VADDR unsigned int
Virtual address of the instruction being executed

ArchMsg.Warning.unpredictable_ici_bits

ICI bits do not apply or are architecturally unpredictable for instruction.

ArchMsg.Warning.unpredictable_t32_breakpoint

Fields:

ADDR1 unsigned int
Address
ADDR2 unsigned int
Breakpoint address
BAS unsigned int
BAS Field
IS_ADDRESS_MISMATCH_BREAKPOINT bool
Is Mismatch breakpoint, not a match breakpoint
N unsigned int
Breakpoint number
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_t32_vector_catch

Fields:

ADDR unsigned int
Address
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_watchpoint_far

Fields:

LOWER_BOUND unsigned int
the lowest address accessed by the instruction that triggered the watchpoint
UPPER_BOUND unsigned int
the highest watchpointed address accessed by that instruction

ArchMsg.Warning.warning_AdvSIMDExpandImmUnexpectedZero

AdvSIMDExpandImm may treat this immediate value as UNPREDICTABLE.

ArchMsg.Warning.warning_ConditionalSMC

SMC instruction has UNPREDICTABLE effects when conditional when combined with traps.

ArchMsg.Warning.warning_access_crosses_page_boundary

Access crosses page boundary.

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault_on_both_pages

Access crosses page boundary has fault(s).

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_access_crosses_page_boundary_spanning_different_memory

Fields:

ADDR unsigned int
: address of access crossing page

MEMTYPE_PAGE1 signed int
: First Page's memory type (0-Normal, 1-Non-Normal)

MEMTYPE_PAGE2 signed int
: Second Page's memory type (0-Normal, 1-Non-Normal)

ArchMsg.Warning.warning_access_wraps_around_memory

Fields:

WIDTH unsigned int
: Address width

ArchMsg.Warning.warning_bx_from_thumbee

Fields:

ADDR unsigned int
: Destination address

ArchMsg.Warning.warning_ccsidr_unimplemented_level

Fields:

CSSELR unsigned int
: current effective value of CSSELR

ArchMsg.Warning.warning_change_to_ns_when_tge_set

Attempting to change NS to 1 when HCR.TGE = 1.

ArchMsg.Warning.warning_contiguous_bit_check_abort

Fields:

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ArchMsg.Warning.warning_contiguous_bit_error

Fields:

CONTIG_ADDR unsigned int
: address of the TLB first read, with which this entry is expected to be contiguous

CONTIG_DATA unsigned int
: expected contents based on the entry at CONTIG_ADDR

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ENTRY_DATA unsigned int
: contents of conflicting TLB entry

ArchMsg.Warning.warning_cp10_cp11_mismatch

Fields:

EL unsigned int
: EL owning the control

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_cp10_cp11_reserved_value

Fields:

CP10 unsigned int
: CP10 access

CP11 unsigned int
: CP11 access

ArchMsg.Warning.warning_csselr_level_out_of_range

Fields:

IMPLEMENTED unsigned int
: number of cache levels implemented

LEVEL unsigned int
: written value of CSSELR.Level

TYPE enum
: The type of cache selected by CSSELR.InD and TnD

ArchMsg.Warning.warning_debug_flow_control_bits_not_obeyed

Fields:

ERROR enum
: type of error resulting

REGISTER enum
: register under consideration

ArchMsg.Warning.warning_debug_register_access_during_reset

Fields:

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

ArchMsg.Warning.warning_decode_cps_inconsistent_fields

The mode setting fields in the CPS instruction are inconsistent.

ArchMsg.Warning.warning_decode_invalid_state

Instruction is unpredictable in current exception-level/security state.

ArchMsg.Warning.warning_default_cacheable_mmu_on

Default cacheable is enabled (HCR.DC = 1) while MMU is enabled.

ArchMsg.Warning.warning_default_cacheable_vmmu_off

Default cacheable is enabled (HCR.DC = 1) while HCR.VM = 0.

ArchMsg.Warning.warning_deprecated_wvr_bit_2_set

Fields:

N unsigned int
: index

WVR unsigned int
: DBGWVR

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_eret_while_software_step_active_pending

Missing ISB between setting MDSCR_EL1.SS and ERET.

ArchMsg.Warning.warning_exclusive_to_non_normal

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_exclusive_to_non_writeback

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_execute_from_device_memory

An attempt was made to execute from Device memory.

ArchMsg.Warning.warning_illegal_cpsr_mode

Fields:

NEW_MODE unsigned int
: New value of CPSR.M

ArchMsg.Warning.warning_illegal_srs_mode

Fields:

MODE unsigned int
: mode for Banked SP

ArchMsg.Warning.warning_implementation_defined_read_debug_register_in_SCS

Read Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_implementation_defined_sequential_security_transitions_supported

The behaviour of a sequential instruction fetches that cross from non-secure to secure memory and contain SG instruction is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_implementation_defined_stack_limit_check_supported

It is IMPLEMENTATION DEFINED whether stack pointer limit checking is performed for the this instructions. Fields:

ADDRESS unsigned int
: Address of instruction

OPCODE unsigned int
: instruction opcode

ArchMsg.Warning.warning_implementation_defined_write_debug_register_in_SCS

Write Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

DATA unsigned int
: data attempted to be written

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PPB_OFFSET unsigned int
debug register offset

ArchMsg.Warning.warning_invalid_tcr_granule

Fields:

REQUEST signed int
page size requested (or 0 for reserved)
SUBSTITUTE signed int
best guess available page size
TG_ID bool
bits TG0 or TG1

ArchMsg.Warning.warning_load_multiple_user_registers_from_user_mode

An LDM(user registers) instruction executed from user mode.

ArchMsg.Warning.warning_load_pc_from_unaligned

PC loaded from an unaligned location.

ArchMsg.Warning.warning_mte_in_mem_attr_not_supported

Fields:

ATTR unsigned int
Memory attribute

ArchMsg.Warning.warning_reserved_breakpoint_state_match

Fields:

IS_BREAKPOINT unsigned int
Is breakpoint, not watchpoint

ArchMsg.Warning.warning_shareability

Unpredictable: combination of the 1st and 2nd stages of translation is Normal Inner Non-Cacheable, Outer Non-Cacheable.

ArchMsg.Warning.warning_software_step_set_while_enabled

MDSCR_EL1.SS set to 1 while software step debug exceptions are enabled.

ArchMsg.Warning.warning_thumb_instruction_wraps_around_memory

Thumb instruction warps around memory.

ArchMsg.Warning.warning_ttbr_sbz_bits_are_not_zero

Fields:

MSB unsigned int
x-1
TTBR enum
which TTBR register

ArchMsg.Warning.warning_unaligned_address_dbgdtrrx_write

Fields:

ADDR unsigned int
Address

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unaligned_address_dbgdtrtx_read

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_unaligned_to_device

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_unaligned_to_strongly_ordered

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_unknown_sau_rnr

SAU_RNR was set to an unsupported value.

ArchMsg.Warning.warning_unpredictable_AIRCR_PRIP_and_BFHFNMINP

The effect of setting both AIRCR.BFHFNMINP and AIRCR.PRIP to 1 is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTCLRACTIVE_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTCLRACTIVE if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_and_SYSRESETREQ

When the processor is halted in Debug state, if a write to the register writes a 1 to both VECTRESET and SYSRESETREQ, the behavior is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTRESET if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_incorrect_VKEY

The value 0x05FA must be written to AIRCR.VKEY, otherwise the register write is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_EXE_RETURN_Reserved_Bit

EXC_RETURN[23:7] are reserved with the special condition that all bits should be written as one. Values other than all 1s are UNPREDICTABLE. Fields:

EXC_RETURN unsigned int
: Special PC value

SBO_MASK unsigned int
: Which bits Should Be One to not UNPRED

ArchMsg.Warning.warning_unpredictable_change_to_DEMCR_MON_STEP_at_insufficient_priority

The effect of changing DEMCR.MON_STEP at an execution priority that is lower than the priority of the DebugMonitor exception is CONSTRAINED UNPREDICTABLE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_change_to_priority_of_active_exception

Changing the priority of an active exception is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_clear_lspact

Fields:

DRVEXC_TAKEN `unsigned int`
: derived exception taken not pended

REACHED_LAST_STORE `unsigned int`
: only the last faulted

ArchMsg.Warning.warning_unpredictable_dbg_exit_unaligned_dlr

Debug exit to AArch32 with DLR[0] set to 1. PC[0] can be set to 0 or to DLR[0].

ArchMsg.Warning.warning_unpredictable_exception_catch

Generation of an exception catch event is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_inconsistent_state

state on exception return is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_instruction

exception return instruction is unpredictable.

ArchMsg.Warning.warning_unpredictable_in_debug_state

Instruction is UNPREDICTABLE when executed in debug state.

ArchMsg.Warning.warning_unpredictable_mair_encoding

Fields:

ATTR `unsigned int`
: Memory attribute

ArchMsg.Warning.warning_unpredictable_pmu_counter_access

Access to PMU counters from non-secure EL0 or EL1 is UNPREDICTABLE with MDCR_EL2.HPMN set to 0.

ArchMsg.Warning.warning_unpredictable_prioritization_breakpoint_match_vector_catch

Two events with the same priority occurred at the same instruction: Address Matching Vector Catch debug event, and Breakpoint debug event It is constrained unpredictable which is taken.

ArchMsg.Warning.warning_unpredictable_stack_selection

UNPREDICTABLE state on context switch.

ArchMsg.Warning.warning_unpredictable_tsize_out_of_range

Fields:

TCR `enum`
: Which TCR

TSIZE `signed int`
: Value of TSize

TSIZE_MAX `signed int`
: The maximum allowed value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

TTBR unsigned int
: Which TTBR region

ArchMsg.Warning.warning_unpredictable_unaligned_pc_as_base_register

The PC value must be word-aligned, otherwise the behavior of the instruction is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_unaligned_pop_stack

Fields:

FRAME_ADDR unsigned int
: unaligned stack frame address

ArchMsg.Warning.warning_unpredictable_vmsa_memattrib

Fields:

TEXCB unsigned int
: texcb value

TEX_REMAP bool
: use tex remap

ArchMsg.Warning.warning_unpredictable_vtcr_t0sz_sl0

Fields:

SL0 signed int
: VTCR.SL0, starting level for stage 2 translation table walks

T0SZ signed int
: VTCR.T0SZ, required input address range

ArchMsg.Warning.warning_unpredictable_write_DHCSR

Fields:

NEW unsigned int
: data attempted to be written

OLD unsigned int
: previous value

ArchMsg.Warning.warning_unpredictable_write_SHCSR

UNPREDICTABLE setting a pending bit to 1 for an exception with priority >= execution priority.

ArchMsg.Warning.warning_unsupported_access_to_memory_mapped_register

Access to memory mapped register is unsupported.

ArchMsg.Warning.warning_user_jmcr_access

Fields:

WRITE unsigned int
: access is write

ArchMsg.Warning.warning_wcr_mask_and_bas

DBGWCR_EL1.MASK is non-zero and BAS is not 0xff.

ArchMsg.Warning.warning_wcr_mask_reserved

DBGWCR_ELn.MASK is set to a reserved value.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_wcr_non_configuous_bas

DBGWCR_EL1.BAS has non-contiguous set of ones.

ArchMsg.Warning.warning_wvr_masked_bit_not_zero

DBGWVRn_EL1.VA contains a bit masked by DBGWCRn_EL1.MASK that is not zero.

ArchMsg.Why.why_branch_target_exception

A branch target exception has occurred. Fields:

BTYPE unsigned int
: pstate.BTYPE at time branch taken

OPCODE unsigned int
: opcode at branch target

PC unsigned int
: Virtual Address of branch target

ArchMsg.Why.why_illegal_state

An illegal mode change has occured. Fields:

MESSAGE string
: The message

BRANCH_MISPREDICT

Simulating branch mispredict. Fields:

PC unsigned int
: Origin address (or 0 if unavailable)

BRA_DIR

Direct branches, to immediate address. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR

Indirect branches, perhaps to a register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR_CCFAIL

Indirect branches fail, perhaps to a register.

BRA_RET_CCFAIL

Branche return fail, perhaps to a register.

CACHE_MAINTENANCE_OP

Cache Maintenance Operation. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Specified MVA or Set/way

FUNCTION enum
: Clean / Invalidate

SCOPE enum
: Affected region

SIDE enum
: Inst / Data.

CCFAIL

Conditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCFAIL_UNC

Unconditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS

Conditional instruction condition check pass. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS_UNC

Unconditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CHECKPOINT_MESSAGE

Report error messages from the checkpointing process. Fields:

message string
: Message contents.

CHECKPOINT_RESTORE_END

Checkpoint restore completed.

CHECKPOINT_RESTORE_START

Checkpoint restore about to start.

CHECKPOINT_SAVE_END

Checkpoint save completed.

CHECKPOINT_SAVE_START

Checkpoint save about to start.

CODE_CACHE_MAINT

Code cache maintenance. Fields:

END_INCL unsigned int
: Inclusive end addres for by VA or PA requests

NS bool
: Non-secure world for PA

START unsigned int
: Start address for by VA or PA requests

TYPE enum
: Request type.

COMPILE_BLOCK_END

Last instruction of basic block translated. Fields:

VADDR unsigned int
: Address of next instruction after this basic block

COMPILE_BLOCK_START

First instruction of basic block translated. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VADDR unsigned int
: Address of first instruction in basic block

COMPILE_INST

ARM instruction compiled. Fields:

DISASS string
: Disassembly of the instruction.

ISET enum
: The instruction set of this instruction.

ITSTATE unsigned int
: The ITSTATE current for the instruction.

OPCODE unsigned int
: The opcode of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

CONTEXTIDR

Context ID Register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

NS bool
: Secure or nonsecure banked register is accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

CONTEXT_SYNC

Called for every context synchronization event. Fields:

ARCH_CSE bool
: Architectural Context Synchronization Event

SYSREG_SYNC bool
: System register synchronization event

CORE_ENDIAN

Core BE8 Big-Endian state changed. Fields:

BE8 bool
: Core BE8 Big-Endian state

CORE_INFO

Static processor attributes. Only triggered by a call to DumpState(). Fields:

ARCH_PROFILE enum
: The architecture profile of the core.

CLUSTER_ID unsigned int
: The cluster ID of this processor.

CORE_NUM unsigned int
: The number of this core in an MP processor.

DETAILED_CORE_FEATURES enum
: Detailed core features as a bitfield. Enum values are bitmasks to extract the info from the field

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

FPU_VERSION enum
: The VFP version implemented by the core.

MEM_ARCH enum
: The memory architecture of the core.

MIDR_EL1 unsigned int
: MIDR_EL1 value

NUM_CORES unsigned int
: The number of cores in this MP processor.

QUANTUM_SIZE unsigned int
: The default quantum size of the core.

SECURITY_FEATURES bool
: Does the core have security features?

CORE_LOADS

Processor load accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_REGS

Changes of the core registers R0 to R14. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number, 0 to 14.

MODE enum
: Bank of the register accessed.

OLD_VALUE unsigned int
: The old value overwritten.

PHYS_ID enum
: The physical register accessed.

VALUE unsigned int
: The new value written to the register.

CORE_REGS64

Changes of the core registers X0..X30, SP_ELn. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: The register number, 0..30 for X0..X30, >=32 for SP_ELn.

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

CORE_STORES

Processor store accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CP14_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP14_WRITE

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

CP15_WRITE

System Control Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

CPSR

CPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

OLD_VALUE unsigned int
: The old CPSR value

UNKNOWN unsigned int
: Bits within the register that have unknown value.

VALUE unsigned int
: The new CPSR value

CRYPTO_SPEC

Every crypto instruction speculatively executed.

CorePowerStateChange

Core power-up state has changed. Fields:

PU bool
: Whether the core powered up

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DATA_CACHE_SET_ALLOC_TAGS

Set allocation tags from DC GVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of memory in bytes.

TAG unsigned int
: The value of the tag being set.

VADDR unsigned int
: Virtual address used for setting allocation tags.

DATA_CACHE_ZERO

Zero and invalidate cache line from DC ZVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of zero'd memory in bytes.

VADDR unsigned int
: Virtual address of the zero'd cache line.

DEBUG_EVENT

Hardware debug support event. Fields:

EVENT enum
: Description of event

VALUE unsigned int
: data value

END_COMPILE

Compilation end. Fields:

END_OF_PAGE_COUNT unsigned int
: Number of basic blocks exited due to page boundary since START_COMPILE.

FETCHFAIL_COUNT unsigned int
: Number of basic blocks exited due to ifetch failure START_COMPILE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: Number of instructions compiled since START_COMPILE.

NONSEQ_COUNT unsigned int
: Number of basic blocks exited due to non-sequential instructions since START_COMPILE.

PAGE_STRADDLE_COUNT unsigned int
: Number of basic blocks exited due to unaligned instructions crossing page since START_COMPILE.

ESB

Error Synchronization Barrier hit. Fields:

ASYNC bool
: ESB is async

IMPLICIT bool
: ESB is implicit

EXCEPTION

Exception taken. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

IS_PHYSICAL bool
: Physical or Virtual exception

LR unsigned int
: The value assigned to the link register.

PC unsigned int
: The location where the exception occurred.

PREFERRED_RETURN unsigned int
: The preferred return address for the exception.

TARGET_ISET enum
: The instruction set of the exception handler code.

TARGET_PC unsigned int
: The address the exception branches to.

VECTOR enum
: The exception vector.

VECTOR_OFFSET unsigned int
: The offset of an exception vector from the base address.

EXCEPTION_END

Every exception completed.

EXCEPTION_RAISE

Every exception raised.

EXCEPTION_RETURN

Branches on leaving exception. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

EXCEPTION_RETURN_PREBRANCH

Exception return event that must occur before PC and processor state have updated.

EXCEPTION_START

Every exception started.

FIQ_TAKEN

FIQ taken exception.

FP_STATE

Floating point state. Fields:

FPSR unsigned int
: FPSR register value.

MASK unsigned int
: The mask indicating updated cumulative bits.

HLT

HLT instruction occurred. Fields:

IMM16 unsigned int
: The 16-bit immediate value encoded in HLT instruction.

INST

Every instruction executed. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DEBUG_STATE bool
: The instruction is executed in debug state.

DISASS string
: Disassembly of instruction.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The current instruction set.

ITSTATE unsigned int
: The current ITSTATE.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

MODE enum
: The mode the core is in.

NS unsigned int
: The core's non-secure bit.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

OPCODE unsigned int
: The opcode of the instruction.

PADDR unsigned int
: The physical address of the instruction.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

INST_END

Every instruction completed.

INST_START

Every instruction started. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

INST_STRADDLE

Instruction straddles boundary. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

IRQ_TAKEN

IRQ taken exception.

ISB

ISB instruction occurred.

LOAD_MULTIPLE_REGS_LIST

List of destination registers (not source) of a multiple registers load instruction. Fields:

REG string
: Register to add into the list

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOCAL_MONITOR

Local monitor activity. Fields:

PADDR unsigned int
: Local Monitor Address

State enum
: State of the monitor (Open/Exclusive)

MEMMAP_DEBUG_READ

Memory mapped reads to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

MEMMAP_DEBUG_WRITE

Memory mapped writes to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

MMU_TRANS

Address translation information. Fields:

ASID unsigned int
: Address space identifier.

CORE_NUM unsigned int
: Core number in a multi processor.

Hyp bool
: Entry matches in Hyp state only

INNERCACHE_RA bool
: Is the inner cache allocate on read

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INNERCACHE_TRANSIENT bool
: Is the inner write-through transient

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

MEMTYPE enum
: Memory type.

NSDESC enum
: Is secure side supposed to access secure or nonsecure memory

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TRANSIENT bool
: Is the outer write-through transient

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

PADDR unsigned int
: Physical address of the access.

PAGESIZE unsigned int
: Page size as log2(size).

SH enum
: Shareability

SIDE enum
: Inst / Data.

STAGE1_PERM unsigned int
: Stage 1 permissions, [5:3] Privileged access (XWR) [2:0] User access (XWR)

STAGE2_PERM unsigned int
: Stage 2 permission mask [5:3] Privileged access (XWR) [2:0] User access (XWR)

VADDR unsigned int
: Virtual address of the access.

VMID unsigned int
: Virtual machine identifier.

XS bool
: XS Attribute

nG enum
: Flag indicating whether ASID will be matched

MMU_TTB_READ

This event is triggered by reads caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data read.

IPA unsigned int
: For stage 1, the IPA of the read.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OUTADDR unsigned int
: The output physical address.

PADDR unsigned int
: The physical address of the read.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MMU_TTB_WRITE

This event is triggered by writes caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data written.

IPA unsigned int
: For stage 1, the IPA of the write.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

PADDR unsigned int
: The physical address of the write.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MODE_CHANGE

Mode change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: The new mode.

NON_SECURE enum
: The new security state

OLD_MODE enum
: The old mode.

NOT_TAG_CHECKED_INST

MTE Tag not checked for current instruction.

PERIODIC

Called for every quantum. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PC unsigned int
The address of the next instruction to be executed on this CPU.

PMU_COUNTER_OVERFLOW

PMU counter overflow. Fields:

EVENT_ID unsigned int
Event Identifier
INDEX unsigned int
Counter index (as selected by PMSELR).
INTERRUPT bool
Is interrupt enabled on overflow for this counter

PREFETCH_MEMORY64

Prefetch from PRFM or PRFUM instructions. Fields:

PRFOP unsigned int
Prefetch hint
VADDR unsigned int
Virtual address of the location that should be prefetched.

PRELOAD_DATA

Data preload from PLD instruction. Fields:

VADDR unsigned int
Virtual address of the data that should be preloaded.

PRELOAD_INST

Instruction preload from PLI instruction. Fields:

VADDR unsigned int
Virtual address of the instruction that should be preloaded.

PRE_CORE_LOAD

Trace just before a core load. Fields:

CURRENT_TIME unsigned int
The core's current time, as simulated time plus local time.
LOCAL_TIME unsigned int
The core's local time, relative to the current quantum.

PRE_CORE_STORE

Trace just before a core store. Fields:

CURRENT_TIME unsigned int
The core's current time, as simulated time plus local time.
LOCAL_TIME unsigned int
The core's local time, relative to the current quantum.

PRE_TTB_READ

This event is triggered before reads caused by a translation table walk. Fields:

ADDR unsigned int
The physical address of the read.

PSTATE_PACTIVE_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions on PChannel. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

PSTATE_PACTIVE_OPMODE_TRANSITION_DENIED

Trace the denied pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_POWER_TRANSITION

Trace pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

PSTATE_PACTIVE_POWER_TRANSITION_DENIED

Trace the denied pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_THREAD_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions for threads on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

RUN_STATE

Run state transition. Fields:

INST_COUNT unsigned int
: Ticks count at point of transition.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NEW enum
: New run state.

OLD enum
: Old run state.

SEMIHOSTING_CALL

Call of a semihost function occurred. Fields:

PC unsigned int
: The program counter after the semihosting call.

SEMIHOSTING_PRECALL

About to call semihost. Fields:

EL2 bool
: Use EL2 translation regime.

NS bool
: Is Non-Secure.

REG_WIDTH unsigned int
: The current register width in bytes.

SIGNAL

External signal state change. Fields:

SIGNAL enum
: Signal that changed

STATE bool
: Signal asserted state

SOFTWARE_STEP

Return the Debug Software Step state. Fields:

STATE enum
: Software Step state

SPSR

SPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: Which of the banked SPSR registers is written.

OLD_VALUE unsigned int
: The old SPSR value

VALUE unsigned int
: The new SPSR value

START_COMPILE

Compilation started. Fields:

VADDR unsigned int
: Instruction where compilation begins.

STG_MULTIPLE_MICRO_OPS_ADDRESS_RANGE

Address range for MTE stores with multiple uops. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

STORE_MULTIPLE_REGS_ADDRESS_RANGE

Address range for stores with multiple registers. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

SUBOPTIMAL_LDST_RETIRED

suboptimal load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1: half-word, 2: word ...)

OPTIMAL_ALIGN_SELECTOR unsigned int
: Selector to decide the optimal alignment size (1: access width, 2: 4-bytes, 3: 8-bytes, ..., 12: 4-kbytes)

VADDR unsigned int
: The virtual address of the access

SYNC

Called for every synchronization, that is every quantum break. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

LOCAL_QUANTUM unsigned int
: The local quantum of this CPU.

LOCAL_TIME unsigned int
: The local time of this CPU.

SYNC_REASON enum
: Reason for sync.

SYSCALL

System call instruction executed. Fields:

IMM unsigned int
: Immediate value of the system call instruction.

TYPE enum
: System call type.

VADDR unsigned int
: Instruction that caused the system call.

SYSREG_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE32

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE64

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

UNALIGNED_LDST_RETIRED

Processor unaligned load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1=>half-word, 2->word ...)

NUMBER_OF_BEATS unsigned int
: Number of accesses (beats) in this burst (Total size in bytes is ACCESS_SIZE * NUMBER_OF_BEATS)

VADDR unsigned int
: The virtual address of the access

VFP_D_REGS

VFP/NEON D 64 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_Q_REGS

VFP/NEON Q 128 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding byte in the register. For example, 0x1 would indicate a write to VALUE[7:0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_SYS_REGS

Writes to the VFP/NEON units system registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: Which VFP system register is written.

MASK unsigned int
: The mask indicating updated cumulative bits.

OLD_VALUE unsigned int
: The register's old value overwritten.

VALUE unsigned int
: The new value written to the VFP system register.

VFP_S_REGS

VFP/NEON S 32 bit register write. Fields:

ALIASING enum
: Type of register aliasing used.

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

WAYPOINT

Signals end of basic block execution. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: Origin instructions set

IS_COND bool
: Indicates if this is a conditional waypoint.

PC unsigned int
: Origin address (or 0 if unavailable)

TAKEN bool
: Indicates if this waypoint was taken.

TARGET unsigned int
: Destination address

TARGET_ISET enum
: Destination instructions set

WFE_END

WFE ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFE.

WFE_EVENT_REGISTER

WFE event register status: set/clear, reason. Fields:

INST_COUNT unsigned int
: Ticks count.

REASON enum
: Reason for set/clear. Only REASON==0 (Cleared by WFE) clears the bit, all other reasons set it.

WFE_IGNORED

WFE ignored. Fields:

EVENT bool
: This WFE was ignored because the event register was set.

INST_COUNT unsigned int
: Ticks count when ignoring WFE.

TRAPPED bool
: This WFE was trapped.

WFE_START

WFE entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFE.

WFI_END

WFI ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFI.

WFI_IGNORED

WFI ignored. Fields:

DISABLED bool
: This WFI was ignored because WFI is disabled.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: Ticks count when ignoring WFI.

REASON enum
: specifies reason why WFI trace was ignored

TRAPPED bool
: This WFI was trapped.

WFI_START

WFI entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFI.

WFI_WAKEUP

WFI wakeup. Fields:

INST_COUNT unsigned int
: Ticks count when WFI wakeup occurred.

REASON enum
: Reason for wakeup.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.14 ARM_Cortex-A5MP - trace

This section describes the trace sources.

ASYNC_MEMORY_FAULT

Context ID Register write. Fields:

FAULT unsigned int
: Fault status in ESR format

PADDR unsigned int
: Physical Address (or 0 if unavailable)

VADDR signed int
: Virtual Address (or 0 if unavailable)

ATOMIC_END_ACCESS

Bus trace access for atomics. Fields:

ACCESS_FAIL bool
: Memory access failed

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

LOAD_VALUE unsigned int
: Loaded values

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ATOMIC_START_ACCESS

Bus trace access for atomics. Fields:

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

NSDESC unsigned int
: The security state of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ArchMsg.Error.exit_code

Fields:

EXITCODE signed int
: Exit code to be returned

KIND string
: Component kind that invoked the exit code trace

REASON string
: Reason why the exit code trace was invoked (optional)

ArchMsg.Warning.TarmacReconstructor

Unexpected event happened during Tarmac reconstruction. Fields:

REASON string
: Reason

ArchMsg.Warning.cache_contents_unknown

A cache has been activated without being invalidated and may contain UNKNOWN content. Fields:

REGIME unsigned int
: Current translation regime

SIDE unsigned int
: 0 d-side cache or unified cache, 1 i-side

ArchMsg.Warning.recursive_branch

An instruction is performing a branch that targets the same instruction. If this was intended then a WFI might be more effective. Fields:

PC unsigned int
: Address of instruction causing the branch

ArchMsg.Warning.recursive_exception

An instruction has generated an exception that targets the same instruction. Fields:

CPSR unsigned int
: Processor state

PC unsigned int
: Address of instruction causing the exception

TYPE enum
: Type of exception

ArchMsg.Warning.tlb_contents_unknown

Translation has been enabled without invalidating the TLB. Fields:

EL unsigned int
: Exception level of translation regime

INVALIDITY unsigned int
: Which TLB may be invalid (if TLBs are separate)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NON_SECURE unsigned int
: Security state of translation regime

VMID unsigned int
: Current VMID, if applicable

BRANCH_MISPREDICT

Simulating branch mispredict. Fields:

PC unsigned int
: Origin address (or 0 if unavailable)

BRA_DIR

Direct branches, to immediate address. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR

Indirect branches, perhaps to a register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR_CCFAIL

Indirect branches fail, perhaps to a register.

BRA_RET_CCFAIL

Branche return fail, perhaps to a register.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CACHE_MAINTENANCE_OP

Cache Maintenance Operation. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Specified MVA or Set/way

FUNCTION enum
: Clean / Invalidate

SCOPE enum
: Affected region

SIDE enum
: Inst / Data.

CCFAIL

Conditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCFAIL_UNC

Unconditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS

Conditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS_UNC

Unconditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CODE_CACHE_MAINT

Code cache maintenance. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

END_INCL unsigned int
: Inclusive end addres for by VA or PA requests

NS bool
: Non-secure world for PA

START unsigned int
: Start address for by VA or PA requests

TYPE enum
: Request type.

COMPILE_BLOCK_END

Last instruction of basic block translated. Fields:

VADDR unsigned int
: Address of next instruction after this basic block

COMPILE_BLOCK_START

First instruction of basic block translated. Fields:

VADDR unsigned int
: Address of first instruction in basic block

COMPILE_INST

ARM instruction compiled. Fields:

DISASS string
: Disassembly of the instruction.

ISET enum
: The instruction set of this instruction.

ITSTATE unsigned int
: The ITSTATE current for the instruction.

OPCODE unsigned int
: The opcode of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

CONTEXTIDR

Context ID Register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

NS bool
: Secure or nonsecure banked register is accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

CONTEXT_SYNC

Called for every context synchronization event. Fields:

ARCH_CSE bool
: Architectural Context Synchronization Event

SYSREG_SYNC bool
: System register synchronization event

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_ENDIAN

Core BE8 Big-Endian state changed. Fields:

BE8 bool
: Core BE8 Big-Endian state

CORE_INFO

Static processor attributes. Only triggered by a call to DumpState(). Fields:

ARCH_PROFILE enum
: The architecture profile of the core.

CLUSTER_ID unsigned int
: The cluster ID of this processor.

CORE_NUM unsigned int
: The number of this core in an MP processor.

DETAILED_CORE_FEATURES enum
: Detailed core features as a bitfield. Enum values are bitmasks to extract the info from the field

FPU_VERSION enum
: The VFP version implemented by the core.

MEM_ARCH enum
: The memory architecture of the core.

MIDR_EL1 unsigned int
: MIDR_EL1 value

NUM_CORES unsigned int
: The number of cores in this MP processor.

QUANTUM_SIZE unsigned int
: The default quantum size of the core.

SECURITY_FEATURES bool
: Does the core have security features?

CORE_LOADS

Processor load accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CORE_REGS

Changes of the core registers R0 to R14. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number, 0 to 14.

MODE enum
: Bank of the register accessed.

OLD_VALUE unsigned int
: The old value overwritten.

PHYS_ID enum
: The physical register accessed.

VALUE unsigned int
: The new value written to the register.

CORE_STORES

Processor store accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CP14_READ

Reads to the System Coprocessor registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP14_WRITE

Writes to the System Coprocessor registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

CP15_WRITE

System Control Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

CPSR

CPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

OLD_VALUE unsigned int
: The old CPSR value

UNKNOWN unsigned int
: Bits within the register that have unknown value.

VALUE unsigned int
: The new CPSR value

CRYPTO_SPEC

Every crypto instruction speculatively executed.

DATA_CACHE_SET_ALLOC_TAGS

Set allocation tags from DC GVA instruction. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of memory in bytes.

TAG unsigned int
: The value of the tag being set.

VADDR unsigned int
: Virtual address used for setting allocation tags.

DATA_CACHE_ZERO

Zero and invalidate cache line from DC ZVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of zero'd memory in bytes.

VADDR unsigned int
: Virtual address of the zero'd cache line.

END_COMPILE

Compilation end. Fields:

END_OF_PAGE_COUNT unsigned int
: Number of basic blocks exited due to page boundary since START_COMPILE.

FETCHFAIL_COUNT unsigned int
: Number of basic blocks exited due to ifetch failure START_COMPILE.

INST_COUNT unsigned int
: Number of instructions compiled since START_COMPILE.

NONSEQ_COUNT unsigned int
: Number of basic blocks exited due to non-sequential instructions since START_COMPILE.

PAGE_STRADDLE_COUNT unsigned int
: Number of basic blocks exited due to unaligned instructions crossing page since START_COMPILE.

EXCEPTION

Exception taken. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

IS_PHYSICAL bool
: Physical or Virtual exception

LR unsigned int
: The value assigned to the link register.

PC unsigned int
: The location where the exception occurred.

PREFERRED_RETURN unsigned int
: The preferred return address for the exception.

TARGET_ISET enum
: The instruction set of the exception handler code.

TARGET_PC unsigned int
: The address the exception branches to.

VECTOR enum
: The exception vector.

VECTOR_OFFSET unsigned int
: The offset of an exception vector from the base address.

EXCEPTION_END

Every exception completed.

EXCEPTION_RAISE

Every exception raised.

EXCEPTION_RETURN

Branches on leaving exception. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

EXCEPTION_RETURN_PREBRANCH

Exception return event that must occur before PC and processor state have updated.

EXCEPTION_START

Every exception started.

FP_STATE

Floating point state. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

FPSR unsigned int
: FPSR register value.

MASK unsigned int
: The mask indicating updated cumulative bits.

HLT

HLT instruction occurred. Fields:

IMM16 unsigned int
: The 16-bit immediate value encoded in HLT instruction.

INST

Every instruction executed. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DEBUG_STATE bool
: The instruction is executed in debug state.

DISASS string
: Disassembly of instruction.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The current instruction set.

ITSTATE unsigned int
: The current ITSTATE.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

MODE enum
: The mode the core is in.

NS unsigned int
: The core's non-secure bit.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

OPCODE unsigned int
: The opcode of the instruction.

PADDR unsigned int
: The physical address of the instruction.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

INST_END

Every instruction completed.

INST_START

Every instruction started. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

INST_STRADDLE

Instruction straddles boundary. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

ISB

ISB instruction occurred.

LOAD_MULTIPLE_REGS_LIST

List of destination registers (not source) of a multiple registers load instruction. Fields:

REG string
: Register to add into the list

LOCAL_MONITOR

Local monitor activity. Fields:

PADDR unsigned int
: Local Monitor Address

State enum
: State of the monitor (Open/Exclusive)

MMU_TRANS

Address translation information. Fields:

ASID unsigned int
: Address space identifier.

CORE_NUM unsigned int
: Core number in a multi processor.

Hyp bool
: Entry matches in Hyp state only

INNERCACHE_RA bool
: Is the inner cache allocate on read

INNERCACHE_TRANSIENT bool
: Is the inner write-through transient

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MEMTYPE enum
: Memory type.

NSDESC enum
: Is secure side supposed to access secure or nonsecure memory

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TRANSIENT bool
: Is the outer write-through transient

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

PADDR unsigned int
: Physical address of the access.

PAGESIZE unsigned int
: Page size as log2(size).

SH enum
: Shareability

SIDE enum
: Inst / Data.

STAGE1_PERM unsigned int
: Stage 1 permissions, [5:3] Privileged access (XWR) [2:0] User access (XWR)

STAGE2_PERM unsigned int
: Stage 2 permission mask [5:3] Privileged access (XWR) [2:0] User access (XWR)

VADDR unsigned int
: Virtual address of the access.

VMID unsigned int
: Virtual machine identifier.

XS bool
: XS Attribute

nG enum
: Flag indicating whether ASID will be matched

MMU_TTB_READ

This event is triggered by reads caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data read.

IPA unsigned int
: For stage 1, the IPA of the read.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

OUTADDR unsigned int
: The output physical address.

PADDR unsigned int
: The physical address of the read.

SIDE enum
: Inst / Data.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

STAGE unsigned int
: Translation stage

MMU_TTB_WRITE

This event is triggered by writes caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data written.

IPA unsigned int
: For stage 1, the IPA of the write.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

PADDR unsigned int
: The physical address of the write.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MODE_CHANGE

Mode change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: The new mode.

NON_SECURE enum
: The new security state

OLD_MODE enum
: The old mode.

NOT_TAG_CHECKED_INST

MTE Tag not checked for current instruction.

PERIODIC

Called for every quantum. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

PC unsigned int
: The address of the next instruction to be executed on this CPU.

PMU_COUNTER_OVERFLOW

PMU counter overflow. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

EVENT_ID unsigned int
: Event Identifier

INDEX unsigned int
: Counter index (as selected by PMSELR).

INTERRUPT bool
: Is interrupt enabled on overflow for this counter

PRELOAD_DATA

Data preload from PLD instruction. Fields:

VADDR unsigned int
: Virtual address of the data that should be preloaded.

PRELOAD_INST

Instruction preload from PLI instruction. Fields:

VADDR unsigned int
: Virtual address of the instruction that should be preloaded.

PRE_CORE_LOAD

Trace just before a core load. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_CORE_STORE

Trace just before a core store. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_TTB_READ

This event is triggered before reads caused by a translation table walk. Fields:

ADDR unsigned int
: The physical address of the read.

PSTATE_PACTIVE_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

PSTATE_PACTIVE_OPMODE_TRANSITION_DENIED

Trace the denied pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_POWER_TRANSITION

Trace pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

PSTATE_PACTIVE_POWER_TRANSITION_DENIED

Trace the denied pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_THREAD_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions for threads on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

RUN_STATE

Run state transition. Fields:

INST_COUNT unsigned int
: Ticks count at point of transition.

NEW enum
: New run state.

OLD enum
: Old run state.

SEMIHOSTING_CALL

Call of a semihost function occurred. Fields:

PC unsigned int
: The program counter after the semihosting call.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SEMIHOSTING_PRECALL

About to call semihost. Fields:

EL2 bool
: Use EL2 translation regime.

NS bool
: Is Non-Secure.

REG_WIDTH unsigned int
: The current register width in bytes.

SIGNAL

External signal state change. Fields:

SIGNAL enum
: Signal that changed

STATE bool
: Signal asserted state

SPSR

SPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: Which of the banked SPSR registers is written.

OLD_VALUE unsigned int
: The old SPSR value

VALUE unsigned int
: The new SPSR value

START_COMPILE

Compilation started. Fields:

VADDR unsigned int
: Instruction where compilation begins.

STG_MULTIPLE_MICRO_OPS_ADDRESS_RANGE

Address range for MTE stores with multiple uops. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

STORE_MULTIPLE_REGS_ADDRESS_RANGE

Address range for stores with multiple registers. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

SUBOPTIMAL_LDST_RETIRED

suboptimal load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1: half-word, 2: word ...)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPTIMAL_ALIGN_SELECTOR unsigned int
: Selector to decide the optimal alignment size (1: access width, 2: 4-bytes, 3: 8-bytes, ..., 12: 4-kbytes)

VADDR unsigned int
: The virtual address of the access

SYNC

Called for every synchronization, that is every quantum break. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

LOCAL_QUANTUM unsigned int
: The local quantum of this CPU.

LOCAL_TIME unsigned int
: The local time of this CPU.

SYNC_REASON enum
: Reason for sync.

SYSCALL

System call instruction executed. Fields:

IMM unsigned int
: Immediate value of the system call instruction.

TYPE enum
: System call type.

VADDR unsigned int
: Instruction that caused the system call.

SYSREG_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SYSREG_UPDATE32

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE64

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

UNALIGNED_LDST_RETIRED

Processor unaligned load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1=>half-word, 2->word ...)

NUMBER_OF_BEATS unsigned int
: Number of accesses (beats) in this burst (Total size in bytes is ACCESS_SIZE * NUMBER_OF_BEATS)

VADDR unsigned int
: The virtual address of the access

VFP_D_REGS

VFP/NEON D 64 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_Q_REGS

VFP/NEON Q 128 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MASK unsigned int
: Each bit indicates a write to the corresponding byte in the register. For example, 0x1 would indicate a write to VALUE[7:0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_SYS_REGS

Writes to the VFP/NEON units system registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: Which VFP system register is written.

MASK unsigned int
: The mask indicating updated cumulative bits.

OLD_VALUE unsigned int
: The register's old value overwritten.

VALUE unsigned int
: The new value written to the VFP system register.

VFP_S_REGS

VFP/NEON S 32 bit register write. Fields:

ALIASING enum
: Type of register aliasing used.

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

WAYPOINT

Signals end of basic block execution. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: Origin instructions set

IS_COND bool
: Indicates if this is a conditional waypoint.

PC unsigned int
: Origin address (or 0 if unavailable)

TAKEN bool
: Indicates if this waypoint was taken.

TARGET unsigned int
: Destination address

TARGET_ISET enum
: Destination instructions set

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

WFE_END

WFE ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFE.

WFE_EVENT_REGISTER

WFE event register status: set/clear, reason. Fields:

INST_COUNT unsigned int
: Ticks count.

REASON enum
: Reason for set/clear. Only REASON==0 (Cleared by WFE) clears the bit, all other reasons set it.

WFE_IGNORED

WFE ignored. Fields:

EVENT bool
: This WFE was ignored because the event register was set.

INST_COUNT unsigned int
: Ticks count when ignoring WFE.

TRAPPED bool
: This WFE was trapped.

WFE_START

WFE entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFE.

WFI_END

WFI ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFI.

WFI_IGNORED

WFI ignored. Fields:

DISABLED bool
: This WFI was ignored because WFI is disabled.

INST_COUNT unsigned int
: Ticks count when ignoring WFI.

REASON enum
: specifies reason why WFI trace was ignored

TRAPPED bool
: This WFI was trapped.

WFI_START

WFI entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFI.

WFI_WAKEUP

WFI wakeup. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: Ticks count when WFI wakeup occurred.

REASON enum
: Reason for wakeup.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.15 ARM_Cortex-A65 - trace

This section describes the trace sources.

ASYNC_MEMORY_FAULT

Context ID Register write. Fields:

FAULT unsigned int
: Fault status in ESR format

PADDR unsigned int
: Physical Address (or 0 if unavailable)

VADDR signed int
: Virtual Address (or 0 if unavailable)

ATOMIC_END_ACCESS

Bus trace access for atomics. Fields:

ACCESS_FAIL bool
: Memory access failed

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

LOAD_VALUE unsigned int
: Loaded values

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ATOMIC_START_ACCESS

Bus trace access for atomics. Fields:

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

NSDESC unsigned int
: The security state of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ArchMsg.Error.exit_code

Fields:

EXITCODE signed int
: Exit code to be returned

KIND string
: Component kind that invoked the exit code trace

REASON string
: Reason why the exit code trace was invoked (optional)

ArchMsg.Warning.IgnoredWarmResetRequest

Event fires when software attempts to request warm reset through write to deprecated DBGPRCR.CWRR.

ArchMsg.Warning.RAS_ErrselrUnpredictableState

ERRSELR is in unpredictable state due to value greater than number_of_error_records written to it.

ArchMsg.Warning.RAS_UnpredictableStateWrite

Fields:

REG_NAME string
: Name of register being written.

ArchMsg.Warning.TarmacReconstructor

Unexpected event happened during Tarmac reconstruction. Fields:

REASON string
: Reason

ArchMsg.Warning.Unpredictably Indexed PM Event Register

Fields:

IsDirect unsigned int
: Direct Access rather than indirect

N unsigned int
: Number of Accessible Registers

SEL unsigned int
: Selector

ArchMsg.Warning.branch_to_unaligned_address

Fields:

ADDR unsigned int
: Unaligned branch address

ArchMsg.Warning.cache_contents_unknown

A cache has been activated without being invalidated and may contain UNKNOWN content. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

REGIME unsigned int
: Current translation regime

SIDE unsigned int
: 0 d-side cache or unified cache, 1 i-side

ArchMsg.Warning.dcimvac_matches_watchpoint

Watchpoints matching an AArch32 DCIMVA are implementation defined.

ArchMsg.Warning.decode_fieldmismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_fracbitsnegative

VCVT Instruction is unpredictable with a negative number of fraction bits.

ArchMsg.Warning.decode_initblock

Instruction is unpredictable in an IT block.

ArchMsg.Warning.decode_initblocknotlast

Instruction is unpredictable in an IT block when not the last.

ArchMsg.Warning.decode_invalid_condition

Instruction is unpredictable when the condition is not AL.

ArchMsg.Warning.decode_invalidfieldcombination

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_invalidvalue

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_mem_hint_unallocated

There is no memory hint allocated to this bit pattern.

ArchMsg.Warning.decode_registermatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_registermismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registeroutofrange

Float point instruction is unpredictable indexing beyond the end of the register bank.

ArchMsg.Warning.decode_sbzsbo

Reserved bits in the instruction are not canonical.

ArchMsg.Warning.decode_transactiontoobig

Attempt to perform a memory transaction of more than 128 bytes.

ArchMsg.Warning.decode_unpred_other

Instruction is unpredictable.

ArchMsg.Warning.decode_unpreduseofpc

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13inreglist

Instruction is unpredictable with R13 in register-list.

ArchMsg.Warning.decode_useofd16orhigher

Attempt to use register D16 or higher where it is not allowed.

ArchMsg.Warning.decode_writebackandbaseinlist

Instruction is unpredictable with write back when the base register is in the transfer list.

ArchMsg.Warning.decode_zeroregistersinlist

Instruction is unpredictable when no registers are to be transfered.

ArchMsg.Warning.recursive_branch

An instruction is performing a branch that targets the same instruction. If this was intended then a WFI might be more effective. Fields:

PC unsigned int
: Address of instruction causing the branch

ArchMsg.Warning.recursive_exception

An instruction has generated an exception that targets the same instruction. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CPSR unsigned int
: Processor state

PC unsigned int
: Address of instruction causing the exception

TYPE enum
: Type of exception

ArchMsg.Warning.reentrant_vector_catch

Fields:

ADDR unsigned int
: vector address

ArchMsg.Warning.reserved_it_state

Fields:

ITSTATE unsigned int
: reserved IT state

ArchMsg.Warning.tlb_contents_unknown

Translation has been enabled without invalidating the TLB. Fields:

EL unsigned int
: Exception level of translation regime

INVALIDITY unsigned int
: Which TLB may be invalid (if TLBs are separate)

NON_SECURE unsigned int
: Security state of translation regime

VMID unsigned int
: Current VMID, if applicable

ArchMsg.Warning.unknown_DLR_DSPSR

Exiting debug state with unknown DLR or DSPSR.

ArchMsg.Warning.unknown_ELR_SPSR

Returning from debug exception with unknown ELR or SPSR.

ArchMsg.Warning.unpredictable_a32_breakpoint

Fields:

ADDR1 unsigned int
: Address

ADDR2 unsigned int
: Breakpoint address

BAS unsigned int
: BAS Field

IS_ADDRESS_MISMATCH_BREAKPOINT bool
: Is Mismatch breakpoint, not a match breakpoint

N unsigned int
: Breakpoint number

OFFSET unsigned int
: Indicates which halfword actually considering into the A32 instruction

ArchMsg.Warning.unpredictable_banked_register_access

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

R unsigned int
R bit
SYSM unsigned int
SYSm

ArchMsg.Warning.unpredictable_ccfail_undef

Fields:

OPCODE unsigned int
opcode of the instruction
VADDR unsigned int
Virtual address of the instruction being executed

ArchMsg.Warning.unpredictable_ici_bits

ICI bits do not apply or are architecturally unpredictable for instruction.

ArchMsg.Warning.unpredictable_t32_breakpoint

Fields:

ADDR1 unsigned int
Address
ADDR2 unsigned int
Breakpoint address
BAS unsigned int
BAS Field
IS_ADDRESS_MISMATCH_BREAKPOINT bool
Is Mismatch breakpoint, not a match breakpoint
N unsigned int
Breakpoint number
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_t32_vector_catch

Fields:

ADDR unsigned int
Address
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_watchpoint_far

Fields:

LOWER_BOUND unsigned int
the lowest address accessed by the instruction that triggered the watchpoint
UPPER_BOUND unsigned int
the highest watchpointed address accessed by that instruction

ArchMsg.Warning.warning_AdvSIMDExpandImmUnexpectedZero

AdvSIMDExpandImm may treat this immediate value as UNPREDICTABLE.

ArchMsg.Warning.warning_ConditionalSMC

SMC instruction has UNPREDICTABLE effects when conditional when combined with traps.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_access_crosses_page_boundary

Access crosses page boundary.

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault_on_both_pages

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_spanning_different_memory

Fields:

ADDR unsigned int
: address of access crossing page

MEMTYPE_PAGE1 signed int
: First Page's memory type (0-Normal, 1-Non-Normal)

MEMTYPE_PAGE2 signed int
: Second Page's memory type (0-Normal, 1-Non-Normal)

ArchMsg.Warning.warning_access_wraps_around_memory

Fields:

WIDTH unsigned int
: Address width

ArchMsg.Warning.warning_bx_from_thumbee

Fields:

ADDR unsigned int
: Destination address

ArchMsg.Warning.warning_ccsidr_unimplemented_level

Fields:

CSSELR unsigned int
: current effective value of CSSELR

ArchMsg.Warning.warning_change_to_ns_when_tge_set

Attempting to change NS to 1 when HCR.TGE = 1.

ArchMsg.Warning.warning_contiguous_bit_check_abort

Fields:

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ArchMsg.Warning.warning_contiguous_bit_error

Fields:

CONTIG_ADDR unsigned int
: address of the TLB first read, with which this entry is expected to be contiguous

CONTIG_DATA unsigned int
: expected contents based on the entry at CONTIG_ADDR

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ENTRY_DATA unsigned int
: contents of conflicting TLB entry

ArchMsg.Warning.warning_cp10_cp11_mismatch

Fields:

EL unsigned int
: EL owning the control

ArchMsg.Warning.warning_cp10_cp11_reserved_value

Fields:

CP10 unsigned int
: CP10 access

CP11 unsigned int
: CP11 access

ArchMsg.Warning.warning_csselr_level_out_of_range

Fields:

IMPLEMENTED unsigned int
: number of cache levels implemented

LEVEL unsigned int
: written value of CSSELR.Level

TYPE enum
: The type of cache selected by CSSELR.InD and TnD

ArchMsg.Warning.warning_debug_flow_control_bits_not_obeyed

Fields:

ERROR enum
: type of error resulting

REGISTER enum
: register under consideration

ArchMsg.Warning.warning_debug_register_access_during_reset

Fields:

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

ArchMsg.Warning.warning_decode_cps_inconsistent_fields

The mode setting fields in the CPS instruction are inconsistent.

ArchMsg.Warning.warning_decode_invalid_state

Instruction is unpredictable in current exception-level/security state.

ArchMsg.Warning.warning_default_cacheable_mmu_on

Default cacheable is enabled (HCR.DC = 1) while MMU is enabled.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_default_cacheable_vmmu_off

Default cacheable is enabled (HCR.DC = 1) while HCR.VM = 0.

ArchMsg.Warning.warning_deprecated_wvr_bit_2_set

Fields:

N unsigned int
: index

WVR unsigned int
: DBGWVR

ArchMsg.Warning.warning_eret_while_software_step_active_pending

Missing ISB between setting MDSCR_EL1.SS and ERET.

ArchMsg.Warning.warning_exclusive_to_non_normal

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_exclusive_to_non_writeback

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_execute_from_device_memory

An attempt was made to execute from Device memory.

ArchMsg.Warning.warning_illegal_cpsr_mode

Fields:

NEW_MODE unsigned int
: New value of CPSR.M

ArchMsg.Warning.warning_illegal_srs_mode

Fields:

MODE unsigned int
: mode for Banked SP

ArchMsg.Warning.warning_implementation_defined_read_debug_register_in_SCS

Read Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_implementation_defined_sequential_security_transitions_supported

The behaviour of a sequential instruction fetches that cross from non-secure to secure memory and contain SG instruction is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_implementation_defined_stack_limit_check_supported

It is IMPLEMENTATION DEFINED whether stack pointer limit checking is performed for the this instructions. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDRESS unsigned int
: Address of instruction

OPCODE unsigned int
: instruction opcode

ArchMsg.Warning.warning_implementation_defined_write_debug_register_in_SCS

Write Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

DATA unsigned int
: data attempted to be written

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_invalid_tcr_granule

Fields:

REQUEST signed int
: page size requested (or 0 for reserved)

SUBSTITUTE signed int
: best guess available page size

TG_ID bool
: bits TG0 or TG1

ArchMsg.Warning.warning_load_multiple_user_registers_from_user_mode

An LDM(user registers) instruction executed from user mode.

ArchMsg.Warning.warning_load_pc_from_unaligned

PC loaded from an unaligned location.

ArchMsg.Warning.warning_mte_in_mem_attr_not_supported

Fields:

ATTR unsigned int
: Memory attribute

ArchMsg.Warning.warning_reserved_breakpoint_state_match

Fields:

IS_BREAKPOINT unsigned int
: Is breakpoint, not watchpoint

ArchMsg.Warning.warning_shareability

Unpredictable: combination of the 1st and 2nd stages of translation is Normal Inner Non-Cacheable, Outer Non-Cacheable.

ArchMsg.Warning.warning_software_step_set_while_enabled

MDSCR_EL1.SS set to 1 while software step debug exceptions are enabled.

ArchMsg.Warning.warning_thumb_instruction_wraps_around_memory

Thumb instruction warps around memory.

ArchMsg.Warning.warning_ttbr_sbz_bits_are_not_zero

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MSB unsigned int
x-1
TTBR enum
which TTBR register

ArchMsg.Warning.warning_unaligned_address_dbgdtrrx_write

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_address_dbgdtrtx_read

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_device

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_strongly_ordered

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unknown_sau_rnr

SAU_RNR was set to an unsupported value.

ArchMsg.Warning.warning_unpredictable_AIRCR_PRIP_and_BFHFNMINP

The effect of setting both AIRCR.BFHFNMINP and AIRCR.PRIP to 1 is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTCLRACTIVE_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTCLRACTIVE if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_and_SYSRESETREQ

When the processor is halted in Debug state, if a write to the register writes a 1 to both VECTRESET and SYSRESETREQ, the behavior is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTRESET if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_incorrect_VKEY

The value 0x05FA must be written to AIRCR.VKEY, otherwise the register write is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_EXE_RETURN_Reserved_Bit

EXC_RETURN[23:7] are reserved with the special condition that all bits should be written as one. Values other than all 1s are UNPREDICTABLE. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

EXC_RETURN unsigned int
: Special PC value

SBO_MASK unsigned int
: Which bits Should Be One to not UNPRED

ArchMsg.Warning.warning_unpredictable_change_to_DEMCR_MON_STEP_at_insufficient_priority

The effect of changing DEMCR.MON_STEP at an execution priority that is lower than the priority of the DebugMonitor exception is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_change_to_priority_of_active_exception

Changing the priority of an active exception is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_clear_lspact

Fields:

DRVEXC_TAKEN unsigned int
: derived exception taken not pended

REACHED_LAST_STORE unsigned int
: only the last faulted

ArchMsg.Warning.warning_unpredictable_dbg_exit_unaligned_dlr

Debug exit to AArch32 with DLR[0] set to 1. PC[0] can be set to 0 or to DLR[0].

ArchMsg.Warning.warning_unpredictable_exception_catch

Generation of an exception catch event is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_inconsistent_state

state on exception return is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_instruction

exception return instruction is unpredictable.

ArchMsg.Warning.warning_unpredictable_in_debug_state

Instruction is UNPREDICTABLE when executed in debug state.

ArchMsg.Warning.warning_unpredictable_mair_encoding

Fields:

ATTR unsigned int
: Memory attribute

ArchMsg.Warning.warning_unpredictable_pmu_counter_access

Access to PMU counters from non-secure EL0 or EL1 is UNPREDICTABLE with MDCR_EL2.HPMN set to 0.

ArchMsg.Warning.warning_unpredictable_prioritization_breakpoint_match_vector_catch

Two events with the same priority occurred at the same instruction: Address Matching Vector Catch debug event, and Breakpoint debug event It is constrained unpredictable which is taken.

ArchMsg.Warning.warning_unpredictable_stack_selection

UNPREDICTABLE state on context switch.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_tsize_out_of_range

Fields:

TCR enum
: Which TCR

TSIZE signed int
: Value of TSize

TSIZE_MAX signed int
: The maximum allowed value

TTBR unsigned int
: Which TTBR region

ArchMsg.Warning.warning_unpredictable_unaligned_pc_as_base_register

The PC value must be word-aligned, otherwise the behavior of the instruction is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_unaligned_pop_stack

Fields:

FRAME_ADDR unsigned int
: unaligned stack frame address

ArchMsg.Warning.warning_unpredictable_vmsa_memattrib

Fields:

TEXCB unsigned int
: texcb value

TEX_REMAP bool
: use tex remap

ArchMsg.Warning.warning_unpredictable_vtcr_t0sz_sl0

Fields:

SL0 signed int
: VTCR.SL0, starting level for stage 2 translation table walks

T0SZ signed int
: VTCR.T0SZ, required input address range

ArchMsg.Warning.warning_unpredictable_write_DHCSR

Fields:

NEW unsigned int
: data attempted to be written

OLD unsigned int
: previous value

ArchMsg.Warning.warning_unpredictable_write_SHCSR

UNPREDICTABLE setting a pending bit to 1 for an exception with priority >= execution priority.

ArchMsg.Warning.warning_unsupported_access_to_memory_mapped_register

Access to memory mapped register is unsupported.

ArchMsg.Warning.warning_user_jmcr_access

Fields:

WRITE unsigned int
: access is write

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_wcr_mask_and_bas

DBGWCR_EL1.MASK is non-zero and BAS is not 0xff.

ArchMsg.Warning.warning_wcr_mask_reserved

DBGWCR_ELn.MASK is set to a reserved value.

ArchMsg.Warning.warning_wcr_non_configuous_bas

DBGWCR_EL1.BAS has non-contiguous set of ones.

ArchMsg.Warning.warning_wvr_masked_bit_not_zero

DBGWVRn_EL1.VA contains a bit masked by DBGWCRn_EL1.MASK that is not zero.

ArchMsg.Why.why_branch_target_exception

A branch target exception has occurred. Fields:

BTYPE unsigned int
: pstate.BTYPE at time branch taken

OPCODE unsigned int
: opcode at branch target

PC unsigned int
: Virtual Address of branch target

ArchMsg.Why.why_illegal_state

An illegal mode change has occured. Fields:

MESSAGE string
: The message

BRANCH_MISPREDICT

Simulating branch mispredict. Fields:

PC unsigned int
: Origin address (or 0 if unavailable)

BRA_DIR

Direct branches, to immediate address. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

BRA_INDIR

Indirect branches, perhaps to a register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR_CCFAIL

Indirect branches fail, perhaps to a register.

BRA_RET_CCFAIL

Branche return fail, perhaps to a register.

CACHE_MAINTENANCE_OP

Cache Maintenance Operation. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Specified MVA or Set/way

FUNCTION enum
: Clean / Invalidate

SCOPE enum
: Affected region

SIDE enum
: Inst / Data.

CCFAIL

Conditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCFAIL_UNC

Unconditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS

Conditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS_UNC

Unconditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CHECKPOINT_MESSAGE

Report error messages from the checkpointing process. Fields:

message string
: Message contents.

CHECKPOINT_RESTORE_END

Checkpoint restore completed.

CHECKPOINT_RESTORE_START

Checkpoint restore about to start.

CHECKPOINT_SAVE_END

Checkpoint save completed.

CHECKPOINT_SAVE_START

Checkpoint save about to start.

CODE_CACHE_MAINT

Code cache maintenance. Fields:

END_INCL unsigned int
: Inclusive end addres for by VA or PA requests

NS bool
: Non-secure world for PA

START unsigned int
: Start address for by VA or PA requests

TYPE enum
: Request type.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

COMPILE_BLOCK_END

Last instruction of basic block translated. Fields:

VADDR unsigned int
: Address of next instruction after this basic block

COMPILE_BLOCK_START

First instruction of basic block translated. Fields:

VADDR unsigned int
: Address of first instruction in basic block

COMPILE_INST

ARM instruction compiled. Fields:

DISASS string
: Disassembly of the instruction.

ISET enum
: The instruction set of this instruction.

ITSTATE unsigned int
: The ITSTATE current for the instruction.

OPCODE unsigned int
: The opcode of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

CONTEXTIDR

Context ID Register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

NS bool
: Secure or nonsecure banked register is accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

CONTEXT_SYNC

Called for every context synchronization event. Fields:

ARCH_CSE bool
: Architectural Context Synchronization Event

SYSREG_SYNC bool
: System register synchronization event

CORE_ENDIAN

Core BE8 Big-Endian state changed. Fields:

BE8 bool
: Core BE8 Big-Endian state

CORE_INFO

Static processor attributes. Only triggered by a call to DumpState(). Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ARCH_PROFILE enum
: The architecture profile of the core.

CLUSTER_ID unsigned int
: The cluster ID of this processor.

CORE_NUM unsigned int
: The number of this core in an MP processor.

DETAILED_CORE_FEATURES enum
: Detailed core features as a bitfield. Enum values are bitmasks to extract the info from the field

FPU_VERSION enum
: The VFP version implemented by the core.

MEM_ARCH enum
: The memory architecture of the core.

MIDR_EL1 unsigned int
: MIDR_EL1 value

NUM_CORES unsigned int
: The number of cores in this MP processor.

QUANTUM_SIZE unsigned int
: The default quantum size of the core.

SECURITY_FEATURES bool
: Does the core have security features?

CORE_LOADS

Processor load accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CORE_REGS

Changes of the core registers R0 to R14. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number, 0 to 14.

MODE enum
: Bank of the register accessed.

OLD_VALUE unsigned int
: The old value overwritten.

PHYS_ID enum
: The physical register accessed.

VALUE unsigned int
: The new value written to the register.

CORE_REGS64

Changes of the core registers X0..X30, SP_ELn. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: The register number, 0..30 for X0..X30, >=32 for SP_ELn.

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

CORE_STORES

Processor store accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CP14_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP14_WRITE

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CP15_WRITE

System Control Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

CPSR

CPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

OLD_VALUE unsigned int
: The old CPSR value

UNKNOWN unsigned int
: Bits within the register that have unknown value.

VALUE unsigned int
: The new CPSR value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CRYPTO_SPEC

Every crypto instruction speculatively executed.

CorePowerStateChange

Core power-up state has changed. Fields:

PU bool
: Whether the core powered up

DATA_CACHE_SET_ALLOC_TAGS

Set allocation tags from DC GVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of memory in bytes.

TAG unsigned int
: The value of the tag being set.

VADDR unsigned int
: Virtual address used for setting allocation tags.

DATA_CACHE_ZERO

Zero and invalidate cache line from DC ZVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of zero'd memory in bytes.

VADDR unsigned int
: Virtual address of the zero'd cache line.

DEBUG_EVENT

Hardware debug support event. Fields:

EVENT enum
: Description of event

VALUE unsigned int
: data value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

END_COMPILE

Compilation end. Fields:

END_OF_PAGE_COUNT unsigned int
: Number of basic blocks exited due to page boundary since START_COMPILE.

FETCHFAIL_COUNT unsigned int
: Number of basic blocks exited due to ifetch failure START_COMPILE.

INST_COUNT unsigned int
: Number of instructions compiled since START_COMPILE.

NONSEQ_COUNT unsigned int
: Number of basic blocks exited due to non-sequential instructions since START_COMPILE.

PAGE_STRADDLE_COUNT unsigned int
: Number of basic blocks exited due to unaligned instructions crossing page since START_COMPILE.

ESB

Error Synchronization Barrier hit. Fields:

ASYNC bool
: ESB is async

IMPLICIT bool
: ESB is implicit

EXCEPTION

Exception taken. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

IS_PHYSICAL bool
: Physical or Virtual exception

LR unsigned int
: The value assigned to the link register.

PC unsigned int
: The location where the exception occurred.

PREFERRED_RETURN unsigned int
: The preferred return address for the exception.

TARGET_ISET enum
: The instruction set of the exception handler code.

TARGET_PC unsigned int
: The address the exception branches to.

VECTOR enum
: The exception vector.

VECTOR_OFFSET unsigned int
: The offset of an exception vector from the base address.

EXCEPTION_END

Every exception completed.

EXCEPTION_RAISE

Every exception raised.

EXCEPTION_RETURN

Branches on leaving exception. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

EXCEPTION_RETURN_PREBRANCH

Exception return event that must occur before PC and processor state have updated.

EXCEPTION_START

Every exception started.

FIQ_TAKEN

FIQ taken exception.

FP_STATE

Floating point state. Fields:

FPSR unsigned int
: FPSR register value.

MASK unsigned int
: The mask indicating updated cumulative bits.

HLT

HLT instruction occurred. Fields:

IMM16 unsigned int
: The 16-bit immediate value encoded in HLT instruction.

INST

Every instruction executed. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DEBUG_STATE bool
: The instruction is executed in debug state.

DISASS string
: Disassembly of instruction.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The current instruction set.

ITSTATE unsigned int
: The current ITSTATE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

MODE enum
: The mode the core is in.

NS unsigned int
: The core's non-secure bit.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

OPCODE unsigned int
: The opcode of the instruction.

PADDR unsigned int
: The physical address of the instruction.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

INST_END

Every instruction completed.

INST_START

Every instruction started. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

INST_STRADDLE

Instruction straddles boundary. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

IRQ_TAKEN

IRQ taken exception.

ISB

ISB instruction occurred.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOAD_MULTIPLE_REGS_LIST

List of destination registers (not source) of a multiple registers load instruction. Fields:

REG string
: Register to add into the list

LOCAL_MONITOR

Local monitor activity. Fields:

PADDR unsigned int
: Local Monitor Address

State enum
: State of the monitor (Open/Exclusive)

MEMMAP_DEBUG_READ

Memory mapped reads to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

MEMMAP_DEBUG_WRITE

Memory mapped writes to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

MMU_TRANS

Address translation information. Fields:

ASID unsigned int
: Address space identifier.

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Hyp bool
: Entry matches in Hyp state only

INNERCACHE_RA bool
: Is the inner cache allocate on read

INNERCACHE_TRANSIENT bool
: Is the inner write-through transient

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

MEMTYPE enum
: Memory type.

NSDESC enum
: Is secure side supposed to access secure or nonsecure memory

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TRANSIENT bool
: Is the outer write-through transient

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

PADDR unsigned int
: Physical address of the access.

PAGESIZE unsigned int
: Page size as log2(size).

SH enum
: Shareability

SIDE enum
: Inst / Data.

STAGE1_PERM unsigned int
: Stage 1 permissions, [5:3] Privileged access (XWR) [2:0] User access (XWR)

STAGE2_PERM unsigned int
: Stage 2 permission mask [5:3] Privileged access (XWR) [2:0] User access (XWR)

VADDR unsigned int
: Virtual address of the access.

VMID unsigned int
: Virtual machine identifier.

XS bool
: XS Attribute

nG enum
: Flag indicating whether ASID will be matched

MMU_TTB_READ

This event is triggered by reads caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data read.

IPA unsigned int
: For stage 1, the IPA of the read.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

OUTADDR unsigned int
: The output physical address.

PADDR unsigned int
: The physical address of the read.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MMU_TTB_WRITE

This event is triggered by writes caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data written.

IPA unsigned int
: For stage 1, the IPA of the write.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

PADDR unsigned int
: The physical address of the write.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MODE_CHANGE

Mode change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: The new mode.

NON_SECURE enum
: The new security state

OLD_MODE enum
: The old mode.

NOT_TAG_CHECKED_INST

MTE Tag not checked for current instruction.

PERIODIC

Called for every quantum. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

PC unsigned int
: The address of the next instruction to be executed on this CPU.

PMU_COUNTER_OVERFLOW

PMU counter overflow. Fields:

EVENT_ID unsigned int
: Event Identifier

INDEX unsigned int
: Counter index (as selected by PMSELR).

INTERRUPT bool
: Is interrupt enabled on overflow for this counter

PREFETCH_MEMORY64

Prefetch from PRFM or PRFUM instructions. Fields:

PRFOP unsigned int
: Prefetch hint

VADDR unsigned int
: Virtual address of the location that should be prefetched.

PRELOAD_DATA

Data preload from PLD instruction. Fields:

VADDR unsigned int
: Virtual address of the data that should be preloaded.

PRELOAD_INST

Instruction preload from PLI instruction. Fields:

VADDR unsigned int
: Virtual address of the instruction that should be preloaded.

PRE_CORE_LOAD

Trace just before a core load. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_CORE_STORE

Trace just before a core store. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_TTB_READ

This event is triggered before reads caused by a translation table walk. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR unsigned int
The physical address of the read.

PSTATE_PACTIVE_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Operating mode after the transition
OLD_STATE enum
Operating mode before the transition

PSTATE_PACTIVE_OPMODE_TRANSITION_DENIED

Trace the denied pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Operating mode after the transition
OLD_STATE enum
Operating mode before the transition
REASON string
Reason for the transition being denied

PSTATE_PACTIVE_POWER_TRANSITION

Trace pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Power mode after the transition
OLD_STATE enum
Power mode before the transition

PSTATE_PACTIVE_POWER_TRANSITION_DENIED

Trace the denied pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Power mode after the transition
OLD_STATE enum
Power mode before the transition
REASON string
Reason for the transition being denied

PSTATE_PACTIVE_THREAD_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions for threads on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Operating mode after the transition
OLD_STATE enum
Operating mode before the transition

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

RUN_STATE

Run state transition. Fields:

INST_COUNT unsigned int
: Ticks count at point of transition.

NEW enum
: New run state.

OLD enum
: Old run state.

SEMIHOSTING_CALL

Call of a semihost function occurred. Fields:

PC unsigned int
: The program counter after the semihosting call.

SEMIHOSTING_PRECALL

About to call semihost. Fields:

EL2 bool
: Use EL2 translation regime.

NS bool
: Is Non-Secure.

REG_WIDTH unsigned int
: The current register width in bytes.

SIGNAL

External signal state change. Fields:

SIGNAL enum
: Signal that changed

STATE bool
: Signal asserted state

SOFTWARE_STEP

Return the Debug Software Step state. Fields:

STATE enum
: Software Step state

SPSR

SPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: Which of the banked SPSR registers is written.

OLD_VALUE unsigned int
: The old SPSR value

VALUE unsigned int
: The new SPSR value

START_COMPILE

Compilation started. Fields:

VADDR unsigned int
: Instruction where compilation begins.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

STG_MULTIPLE_MICRO_OPS_ADDRESS_RANGE

Address range for MTE stores with multiple uops. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

STORE_MULTIPLE_REGS_ADDRESS_RANGE

Address range for stores with multiple registers. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

SUBOPTIMAL_LDST_RETIRED

suboptimal load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1: half-word, 2: word ...)

OPTIMAL_ALIGN_SELECTOR unsigned int
: Selector to decide the optimal alignment size (1: access width, 2: 4-bytes, 3: 8-bytes, ..., 12: 4-kbytes)

VADDR unsigned int
: The virtual address of the access

SYNC

Called for every synchronization, that is every quantum break. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

LOCAL_QUANTUM unsigned int
: The local quantum of this CPU.

LOCAL_TIME unsigned int
: The local time of this CPU.

SYNC_REASON enum
: Reason for sync.

SYSCALL

System call instruction executed. Fields:

IMM unsigned int
: Immediate value of the system call instruction.

TYPE enum
: System call type.

VADDR unsigned int
: Instruction that caused the system call.

SYSREG_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE32

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE64

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

UNALIGNED_LDST_RETIRED

Processor unaligned load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1=>half-word, 2->word ...)

NUMBER_OF_BEATS unsigned int
: Number of accesses (beats) in this burst (Total size in bytes is ACCESS_SIZE * NUMBER_OF_BEATS)

VADDR unsigned int
: The virtual address of the access

VFP_D_REGS

VFP/NEON D 64 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_Q_REGS

VFP/NEON Q 128 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding byte in the register. For example, 0x1 would indicate a write to VALUE[7:0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_SYS_REGS

Writes to the VFP/NEON units system registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: Which VFP system register is written.

MASK unsigned int
: The mask indicating updated cumulative bits.

OLD_VALUE unsigned int
: The register's old value overwritten.

VALUE unsigned int
: The new value written to the VFP system register.

VFP_S_REGS

VFP/NEON S 32 bit register write. Fields:

ALIASING enum
: Type of register aliasing used.

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VALUE unsigned int
: The new value written to the register.

WAYPOINT

Signals end of basic block execution. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: Origin instructions set

IS_COND bool
: Indicates if this is a conditional waypoint.

PC unsigned int
: Origin address (or 0 if unavailable)

TAKEN bool
: Indicates if this waypoint was taken.

TARGET unsigned int
: Destination address

TARGET_ISET enum
: Destination instructions set

WFE_END

WFE ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFE.

WFE_EVENT_REGISTER

WFE event register status: set/clear, reason. Fields:

INST_COUNT unsigned int
: Ticks count.

REASON enum
: Reason for set/clear. Only REASON==0 (Cleared by WFE) clears the bit, all other reasons set it.

WFE_IGNORED

WFE ignored. Fields:

EVENT bool
: This WFE was ignored because the event register was set.

INST_COUNT unsigned int
: Ticks count when ignoring WFE.

TRAPPED bool
: This WFE was trapped.

WFE_START

WFE entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFE.

WFI_END

WFI ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFI.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

WFI_IGNORED

WFI ignored. Fields:

DISABLED bool
: This WFI was ignored because WFI is disabled.

INST_COUNT unsigned int
: Ticks count when ignoring WFI.

REASON enum
: specifies reason why WFI trace was ignored

TRAPPED bool
: This WFI was trapped.

WFI_START

WFI entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFI.

WFI_WAKEUP

WFI wakeup. Fields:

INST_COUNT unsigned int
: Ticks count when WFI wakeup occurred.

REASON enum
: Reason for wakeup.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.16 ARM_Cortex-A65AE - trace

This section describes the trace sources.

ASYNC_MEMORY_FAULT

Context ID Register write. Fields:

FAULT unsigned int
: Fault status in ESR format

PADDR unsigned int
: Physical Address (or 0 if unavailable)

VADDR signed int
: Virtual Address (or 0 if unavailable)

ATOMIC_END_ACCESS

Bus trace access for atomics. Fields:

ACCESS_FAIL bool
: Memory access failed

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

LOAD_VALUE unsigned int
: Loaded values

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ATOMIC_START_ACCESS

Bus trace access for atomics. Fields:

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

NSDESC unsigned int
: The security state of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ArchMsg.Error.exit_code

Fields:

EXITCODE signed int
: Exit code to be returned

KIND string
: Component kind that invoked the exit code trace

REASON string
: Reason why the exit code trace was invoked (optional)

ArchMsg.Warning.IgnoredWarmResetRequest

Event fires when software attempts to request warm reset through write to deprecated DBGPRCR.CWRR.

ArchMsg.Warning.RAS_ErrselrUnpredictableState

ERRSELR is in unpredictable state due to value greater than number_of_error_records written to it.

ArchMsg.Warning.RAS_UnpredictableStateWrite

Fields:

REG_NAME string
: Name of register being written.

ArchMsg.Warning.TarmacReconstructor

Unexpected event happened during Tarmac reconstruction. Fields:

REASON string
: Reason

ArchMsg.Warning.Unpredictably Indexed PM Event Register

Fields:

IsDirect unsigned int
: Direct Access rather than indirect

N unsigned int
: Number of Accessible Registers

SEL unsigned int
: Selector

ArchMsg.Warning.branch_to_unaligned_address

Fields:

ADDR unsigned int
: Unaligned branch address

ArchMsg.Warning.cache_contents_unknown

A cache has been activated without being invalidated and may contain UNKNOWN content. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

REGIME unsigned int
: Current translation regime

SIDE unsigned int
: 0 d-side cache or unified cache, 1 i-side

ArchMsg.Warning.dcimvac_matches_watchpoint

Watchpoints matching an AArch32 DCIMVA are implementation defined.

ArchMsg.Warning.decode_fieldmismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_fracbitsnegative

VCVT Instruction is unpredictable with a negative number of fraction bits.

ArchMsg.Warning.decode_initblock

Instruction is unpredictable in an IT block.

ArchMsg.Warning.decode_initblocknotlast

Instruction is unpredictable in an IT block when not the last.

ArchMsg.Warning.decode_invalid_condition

Instruction is unpredictable when the condition is not AL.

ArchMsg.Warning.decode_invalidfieldcombination

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_invalidvalue

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_mem_hint_unallocated

There is no memory hint allocated to this bit pattern.

ArchMsg.Warning.decode_registermatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_registermismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registeroutofrange

Float point instruction is unpredictable indexing beyond the end of the register bank.

ArchMsg.Warning.decode_sbzsbo

Reserved bits in the instruction are not canonical.

ArchMsg.Warning.decode_transactiontoobig

Attempt to perform a memory transaction of more than 128 bytes.

ArchMsg.Warning.decode_unpred_other

Instruction is unpredictable.

ArchMsg.Warning.decode_unpreduseofpc

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13inreglist

Instruction is unpredictable with R13 in register-list.

ArchMsg.Warning.decode_useofd16orhigher

Attempt to use register D16 or higher where it is not allowed.

ArchMsg.Warning.decode_writebackandbaseinlist

Instruction is unpredictable with write back when the base register is in the transfer list.

ArchMsg.Warning.decode_zeroregistersinlist

Instruction is unpredictable when no registers are to be transfered.

ArchMsg.Warning.recursive_branch

An instruction is performing a branch that targets the same instruction. If this was intended then a WFI might be more effective. Fields:

PC unsigned int
: Address of instruction causing the branch

ArchMsg.Warning.recursive_exception

An instruction has generated an exception that targets the same instruction. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CPSR unsigned int
: Processor state

PC unsigned int
: Address of instruction causing the exception

TYPE enum
: Type of exception

ArchMsg.Warning.reentrant_vector_catch

Fields:

ADDR unsigned int
: vector address

ArchMsg.Warning.reserved_it_state

Fields:

ITSTATE unsigned int
: reserved IT state

ArchMsg.Warning.tlb_contents_unknown

Translation has been enabled without invalidating the TLB. Fields:

EL unsigned int
: Exception level of translation regime

INVALIDITY unsigned int
: Which TLB may be invalid (if TLBs are separate)

NON_SECURE unsigned int
: Security state of translation regime

VMID unsigned int
: Current VMID, if applicable

ArchMsg.Warning.unknown_DLR_DSPSR

Exiting debug state with unknown DLR or DSPSR.

ArchMsg.Warning.unknown_ELR_SPSR

Returning from debug exception with unknown ELR or SPSR.

ArchMsg.Warning.unpredictable_a32_breakpoint

Fields:

ADDR1 unsigned int
: Address

ADDR2 unsigned int
: Breakpoint address

BAS unsigned int
: BAS Field

IS_ADDRESS_MISMATCH_BREAKPOINT bool
: Is Mismatch breakpoint, not a match breakpoint

N unsigned int
: Breakpoint number

OFFSET unsigned int
: Indicates which halfword actually considering into the A32 instruction

ArchMsg.Warning.unpredictable_banked_register_access

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

R unsigned int
R bit
SYSM unsigned int
SYSm

ArchMsg.Warning.unpredictable_ccfail_undef

Fields:

OPCODE unsigned int
opcode of the instruction
VADDR unsigned int
Virtual address of the instruction being executed

ArchMsg.Warning.unpredictable_ici_bits

ICI bits do not apply or are architecturally unpredictable for instruction.

ArchMsg.Warning.unpredictable_t32_breakpoint

Fields:

ADDR1 unsigned int
Address
ADDR2 unsigned int
Breakpoint address
BAS unsigned int
BAS Field
IS_ADDRESS_MISMATCH_BREAKPOINT bool
Is Mismatch breakpoint, not a match breakpoint
N unsigned int
Breakpoint number
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_t32_vector_catch

Fields:

ADDR unsigned int
Address
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_watchpoint_far

Fields:

LOWER_BOUND unsigned int
the lowest address accessed by the instruction that triggered the watchpoint
UPPER_BOUND unsigned int
the highest watchpointed address accessed by that instruction

ArchMsg.Warning.warning_AdvSIMDExpandImmUnexpectedZero

AdvSIMDExpandImm may treat this immediate value as UNPREDICTABLE.

ArchMsg.Warning.warning_ConditionalSMC

SMC instruction has UNPREDICTABLE effects when conditional when combined with traps.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_access_crosses_page_boundary

Access crosses page boundary.

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault_on_both_pages

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_spanning_different_memory

Fields:

ADDR unsigned int
: address of access crossing page

MEMTYPE_PAGE1 signed int
: First Page's memory type (0-Normal, 1-Non-Normal)

MEMTYPE_PAGE2 signed int
: Second Page's memory type (0-Normal, 1-Non-Normal)

ArchMsg.Warning.warning_access_wraps_around_memory

Fields:

WIDTH unsigned int
: Address width

ArchMsg.Warning.warning_bx_from_thumbee

Fields:

ADDR unsigned int
: Destination address

ArchMsg.Warning.warning_ccsidr_unimplemented_level

Fields:

CSSELR unsigned int
: current effective value of CSSELR

ArchMsg.Warning.warning_change_to_ns_when_tge_set

Attempting to change NS to 1 when HCR.TGE = 1.

ArchMsg.Warning.warning_contiguous_bit_check_abort

Fields:

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ArchMsg.Warning.warning_contiguous_bit_error

Fields:

CONTIG_ADDR unsigned int
: address of the TLB first read, with which this entry is expected to be contiguous

CONTIG_DATA unsigned int
: expected contents based on the entry at CONTIG_ADDR

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ENTRY_DATA unsigned int
: contents of conflicting TLB entry

ArchMsg.Warning.warning_cp10_cp11_mismatch

Fields:

EL unsigned int
: EL owning the control

ArchMsg.Warning.warning_cp10_cp11_reserved_value

Fields:

CP10 unsigned int
: CP10 access

CP11 unsigned int
: CP11 access

ArchMsg.Warning.warning_csselr_level_out_of_range

Fields:

IMPLEMENTED unsigned int
: number of cache levels implemented

LEVEL unsigned int
: written value of CSSELR.Level

TYPE enum
: The type of cache selected by CSSELR.InD and TnD

ArchMsg.Warning.warning_debug_flow_control_bits_not_obeyed

Fields:

ERROR enum
: type of error resulting

REGISTER enum
: register under consideration

ArchMsg.Warning.warning_debug_register_access_during_reset

Fields:

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

ArchMsg.Warning.warning_decode_cps_inconsistent_fields

The mode setting fields in the CPS instruction are inconsistent.

ArchMsg.Warning.warning_decode_invalid_state

Instruction is unpredictable in current exception-level/security state.

ArchMsg.Warning.warning_default_cacheable_mmu_on

Default cacheable is enabled (HCR.DC = 1) while MMU is enabled.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_default_cacheable_vmmu_off

Default cacheable is enabled (HCR.DC = 1) while HCR.VM = 0.

ArchMsg.Warning.warning_deprecated_wvr_bit_2_set

Fields:

N unsigned int
: index

WVR unsigned int
: DBGWVR

ArchMsg.Warning.warning_eret_while_software_step_active_pending

Missing ISB between setting MDSCR_EL1.SS and ERET.

ArchMsg.Warning.warning_exclusive_to_non_normal

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_exclusive_to_non_writeback

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_execute_from_device_memory

An attempt was made to execute from Device memory.

ArchMsg.Warning.warning_illegal_cpsr_mode

Fields:

NEW_MODE unsigned int
: New value of CPSR.M

ArchMsg.Warning.warning_illegal_srs_mode

Fields:

MODE unsigned int
: mode for Banked SP

ArchMsg.Warning.warning_implementation_defined_read_debug_register_in_SCS

Read Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_implementation_defined_sequential_security_transitions_supported

The behaviour of a sequential instruction fetches that cross from non-secure to secure memory and contain SG instruction is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_implementation_defined_stack_limit_check_supported

It is IMPLEMENTATION DEFINED whether stack pointer limit checking is performed for the this instructions. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDRESS unsigned int
: Address of instruction

OPCODE unsigned int
: instruction opcode

ArchMsg.Warning.warning_implementation_defined_write_debug_register_in_SCS

Write Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

DATA unsigned int
: data attempted to be written

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_invalid_tcr_granule

Fields:

REQUEST signed int
: page size requested (or 0 for reserved)

SUBSTITUTE signed int
: best guess available page size

TG_ID bool
: bits TG0 or TG1

ArchMsg.Warning.warning_load_multiple_user_registers_from_user_mode

An LDM(user registers) instruction executed from user mode.

ArchMsg.Warning.warning_load_pc_from_unaligned

PC loaded from an unaligned location.

ArchMsg.Warning.warning_mte_in_mem_attr_not_supported

Fields:

ATTR unsigned int
: Memory attribute

ArchMsg.Warning.warning_reserved_breakpoint_state_match

Fields:

IS_BREAKPOINT unsigned int
: Is breakpoint, not watchpoint

ArchMsg.Warning.warning_shareability

Unpredictable: combination of the 1st and 2nd stages of translation is Normal Inner Non-Cacheable, Outer Non-Cacheable.

ArchMsg.Warning.warning_software_step_set_while_enabled

MDSCR_EL1.SS set to 1 while software step debug exceptions are enabled.

ArchMsg.Warning.warning_thumb_instruction_wraps_around_memory

Thumb instruction warps around memory.

ArchMsg.Warning.warning_ttbr_sbz_bits_are_not_zero

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MSB unsigned int
: x-1

TTBR enum
: which TTBR register

ArchMsg.Warning.warning_unaligned_address_dbgdtrrx_write

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_unaligned_address_dbgdtrtx_read

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_unaligned_to_device

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_unaligned_to_strongly_ordered

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_unknown_sau_rnr

SAU_RNR was set to an unsupported value.

ArchMsg.Warning.warning_unpredictable_AIRCR_PRIP_and_BFHFNMINP

The effect of setting both AIRCR.BFHFNMINP and AIRCR.PRIP to 1 is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTCLRACTIVE_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTCLRACTIVE if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_and_SYSRESETREQ

When the processor is halted in Debug state, if a write to the register writes a 1 to both VECTRESET and SYSRESETREQ, the behavior is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTRESET if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_incorrect_VKEY

The value 0x05FA must be written to AIRCR.VKEY, otherwise the register write is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_EXE_RETURN_Reserved_Bit

EXC_RETURN[23:7] are reserved with the special condition that all bits should be written as one. Values other than all 1s are UNPREDICTABLE. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

EXC_RETURN unsigned int
Special PC value
SBO_MASK unsigned int
Which bits Should Be One to not UNPRED

ArchMsg.Warning.warning_unpredictable_change_to_DEMCR_MON_STEP_at_insufficient_priority

The effect of changing DEMCR.MON_STEP at an execution priority that is lower than the priority of the DebugMonitor exception is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_change_to_priority_of_active_exception

Changing the priority of an active exception is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_clear_lspact

Fields:

DRVEXC_TAKEN unsigned int
derived exception taken not pended
REACHED_LAST_STORE unsigned int
only the last faulted

ArchMsg.Warning.warning_unpredictable_dbg_exit_unaligned_dlr

Debug exit to AArch32 with DLR[0] set to 1. PC[0] can be set to 0 or to DLR[0].

ArchMsg.Warning.warning_unpredictable_exception_catch

Generation of an exception catch event is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_inconsistent_state

state on exception return is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_instruction

exception return instruction is unpredictable.

ArchMsg.Warning.warning_unpredictable_in_debug_state

Instruction is UNPREDICTABLE when executed in debug state.

ArchMsg.Warning.warning_unpredictable_mair_encoding

Fields:

ATTR unsigned int
Memory attribute

ArchMsg.Warning.warning_unpredictable_pmu_counter_access

Access to PMU counters from non-secure EL0 or EL1 is UNPREDICTABLE with MDCR_EL2.HPMN set to 0.

ArchMsg.Warning.warning_unpredictable_prioritization_breakpoint_match_vector_catch

Two events with the same priority occurred at the same instruction: Address Matching Vector Catch debug event, and Breakpoint debug event It is constrained unpredictable which is taken.

ArchMsg.Warning.warning_unpredictable_stack_selection

UNPREDICTABLE state on context switch.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_tsize_out_of_range

Fields:

TCR enum
: Which TCR

TSIZE signed int
: Value of TSize

TSIZE_MAX signed int
: The maximum allowed value

TTBR unsigned int
: Which TTBR region

ArchMsg.Warning.warning_unpredictable_unaligned_pc_as_base_register

The PC value must be word-aligned, otherwise the behavior of the instruction is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_unaligned_pop_stack

Fields:

FRAME_ADDR unsigned int
: unaligned stack frame address

ArchMsg.Warning.warning_unpredictable_vmsa_memattrib

Fields:

TEXCB unsigned int
: texcb value

TEX_REMAP bool
: use tex remap

ArchMsg.Warning.warning_unpredictable_vtcr_t0sz_sl0

Fields:

SL0 signed int
: VTCR.SL0, starting level for stage 2 translation table walks

T0SZ signed int
: VTCR.T0SZ, required input address range

ArchMsg.Warning.warning_unpredictable_write_DHCSR

Fields:

NEW unsigned int
: data attempted to be written

OLD unsigned int
: previous value

ArchMsg.Warning.warning_unpredictable_write_SHCSR

UNPREDICTABLE setting a pending bit to 1 for an exception with priority >= execution priority.

ArchMsg.Warning.warning_unsupported_access_to_memory_mapped_register

Access to memory mapped register is unsupported.

ArchMsg.Warning.warning_user_jmcr_access

Fields:

WRITE unsigned int
: access is write

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_wcr_mask_and_bas

DBGWCR_EL1.MASK is non-zero and BAS is not 0xff.

ArchMsg.Warning.warning_wcr_mask_reserved

DBGWCR_ELn.MASK is set to a reserved value.

ArchMsg.Warning.warning_wcr_non_configuous_bas

DBGWCR_EL1.BAS has non-contiguous set of ones.

ArchMsg.Warning.warning_wvr_masked_bit_not_zero

DBGWVRn_EL1.VA contains a bit masked by DBGWCRn_EL1.MASK that is not zero.

ArchMsg.Why.why_branch_target_exception

A branch target exception has occurred. Fields:

BTYPE unsigned int
: pstate.BTYPE at time branch taken

OPCODE unsigned int
: opcode at branch target

PC unsigned int
: Virtual Address of branch target

ArchMsg.Why.why_illegal_state

An illegal mode change has occured. Fields:

MESSAGE string
: The message

BRANCH_MISPREDICT

Simulating branch mispredict. Fields:

PC unsigned int
: Origin address (or 0 if unavailable)

BRA_DIR

Direct branches, to immediate address. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

BRA_INDIR

Indirect branches, perhaps to a register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR_CCFAIL

Indirect branches fail, perhaps to a register.

BRA_RET_CCFAIL

Branche return fail, perhaps to a register.

CACHE_MAINTENANCE_OP

Cache Maintenance Operation. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Specified MVA or Set/way

FUNCTION enum
: Clean / Invalidate

SCOPE enum
: Affected region

SIDE enum
: Inst / Data.

CCFAIL

Conditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCFAIL_UNC

Unconditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS

Conditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS_UNC

Unconditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CHECKPOINT_MESSAGE

Report error messages from the checkpointing process. Fields:

message string
: Message contents.

CHECKPOINT_RESTORE_END

Checkpoint restore completed.

CHECKPOINT_RESTORE_START

Checkpoint restore about to start.

CHECKPOINT_SAVE_END

Checkpoint save completed.

CHECKPOINT_SAVE_START

Checkpoint save about to start.

CODE_CACHE_MAINT

Code cache maintenance. Fields:

END_INCL unsigned int
: Inclusive end addres for by VA or PA requests

NS bool
: Non-secure world for PA

START unsigned int
: Start address for by VA or PA requests

TYPE enum
: Request type.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

COMPILE_BLOCK_END

Last instruction of basic block translated. Fields:

VADDR unsigned int
: Address of next instruction after this basic block

COMPILE_BLOCK_START

First instruction of basic block translated. Fields:

VADDR unsigned int
: Address of first instruction in basic block

COMPILE_INST

ARM instruction compiled. Fields:

DISASS string
: Disassembly of the instruction.

ISET enum
: The instruction set of this instruction.

ITSTATE unsigned int
: The ITSTATE current for the instruction.

OPCODE unsigned int
: The opcode of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

CONTEXTIDR

Context ID Register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

NS bool
: Secure or nonsecure banked register is accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

CONTEXT_SYNC

Called for every context synchronization event. Fields:

ARCH_CSE bool
: Architectural Context Synchronization Event

SYSREG_SYNC bool
: System register synchronization event

CORE_ENDIAN

Core BE8 Big-Endian state changed. Fields:

BE8 bool
: Core BE8 Big-Endian state

CORE_INFO

Static processor attributes. Only triggered by a call to DumpState(). Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ARCH_PROFILE enum
: The architecture profile of the core.

CLUSTER_ID unsigned int
: The cluster ID of this processor.

CORE_NUM unsigned int
: The number of this core in an MP processor.

DETAILED_CORE_FEATURES enum
: Detailed core features as a bitfield. Enum values are bitmasks to extract the info from the field

FPU_VERSION enum
: The VFP version implemented by the core.

MEM_ARCH enum
: The memory architecture of the core.

MIDR_EL1 unsigned int
: MIDR_EL1 value

NUM_CORES unsigned int
: The number of cores in this MP processor.

QUANTUM_SIZE unsigned int
: The default quantum size of the core.

SECURITY_FEATURES bool
: Does the core have security features?

CORE_LOADS

Processor load accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CORE_REGS

Changes of the core registers R0 to R14. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number, 0 to 14.

MODE enum
: Bank of the register accessed.

OLD_VALUE unsigned int
: The old value overwritten.

PHYS_ID enum
: The physical register accessed.

VALUE unsigned int
: The new value written to the register.

CORE_REGS64

Changes of the core registers X0..X30, SP_ELn. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: The register number, 0..30 for X0..X30, >=32 for SP_ELn.

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

CORE_STORES

Processor store accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CP14_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP14_WRITE

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CP15_WRITE

System Control Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

CPSR

CPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

OLD_VALUE unsigned int
: The old CPSR value

UNKNOWN unsigned int
: Bits within the register that have unknown value.

VALUE unsigned int
: The new CPSR value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CRYPTO_SPEC

Every crypto instruction speculatively executed.

CorePowerStateChange

Core power-up state has changed. Fields:

PU bool
: Whether the core powered up

DATA_CACHE_SET_ALLOC_TAGS

Set allocation tags from DC GVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of memory in bytes.

TAG unsigned int
: The value of the tag being set.

VADDR unsigned int
: Virtual address used for setting allocation tags.

DATA_CACHE_ZERO

Zero and invalidate cache line from DC ZVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of zero'd memory in bytes.

VADDR unsigned int
: Virtual address of the zero'd cache line.

DEBUG_EVENT

Hardware debug support event. Fields:

EVENT enum
: Description of event

VALUE unsigned int
: data value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

END_COMPILE

Compilation end. Fields:

END_OF_PAGE_COUNT unsigned int
: Number of basic blocks exited due to page boundary since START_COMPILE.

FETCHFAIL_COUNT unsigned int
: Number of basic blocks exited due to ifetch failure START_COMPILE.

INST_COUNT unsigned int
: Number of instructions compiled since START_COMPILE.

NONSEQ_COUNT unsigned int
: Number of basic blocks exited due to non-sequential instructions since START_COMPILE.

PAGE_STRADDLE_COUNT unsigned int
: Number of basic blocks exited due to unaligned instructions crossing page since START_COMPILE.

ESB

Error Synchronization Barrier hit. Fields:

ASYNC bool
: ESB is async

IMPLICIT bool
: ESB is implicit

EXCEPTION

Exception taken. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

IS_PHYSICAL bool
: Physical or Virtual exception

LR unsigned int
: The value assigned to the link register.

PC unsigned int
: The location where the exception occurred.

PREFERRED_RETURN unsigned int
: The preferred return address for the exception.

TARGET_ISET enum
: The instruction set of the exception handler code.

TARGET_PC unsigned int
: The address the exception branches to.

VECTOR enum
: The exception vector.

VECTOR_OFFSET unsigned int
: The offset of an exception vector from the base address.

EXCEPTION_END

Every exception completed.

EXCEPTION_RAISE

Every exception raised.

EXCEPTION_RETURN

Branches on leaving exception. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

EXCEPTION_RETURN_PREBRANCH

Exception return event that must occur before PC and processor state have updated.

EXCEPTION_START

Every exception started.

FIQ_TAKEN

FIQ taken exception.

FP_STATE

Floating point state. Fields:

FPSR unsigned int
: FPSR register value.

MASK unsigned int
: The mask indicating updated cumulative bits.

HLT

HLT instruction occurred. Fields:

IMM16 unsigned int
: The 16-bit immediate value encoded in HLT instruction.

INST

Every instruction executed. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DEBUG_STATE bool
: The instruction is executed in debug state.

DISASS string
: Disassembly of instruction.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The current instruction set.

ITSTATE unsigned int
: The current ITSTATE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

MODE enum
: The mode the core is in.

NS unsigned int
: The core's non-secure bit.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

OPCODE unsigned int
: The opcode of the instruction.

PADDR unsigned int
: The physical address of the instruction.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

INST_END

Every instruction completed.

INST_START

Every instruction started. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

INST_STRADDLE

Instruction straddles boundary. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

IRQ_TAKEN

IRQ taken exception.

ISB

ISB instruction occurred.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOAD_MULTIPLE_REGS_LIST

List of destination registers (not source) of a multiple registers load instruction. Fields:

REG string
: Register to add into the list

LOCAL_MONITOR

Local monitor activity. Fields:

PADDR unsigned int
: Local Monitor Address

State enum
: State of the monitor (Open/Exclusive)

MEMMAP_DEBUG_READ

Memory mapped reads to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

MEMMAP_DEBUG_WRITE

Memory mapped writes to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

MMU_TRANS

Address translation information. Fields:

ASID unsigned int
: Address space identifier.

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Hyp bool
: Entry matches in Hyp state only

INNERCACHE_RA bool
: Is the inner cache allocate on read

INNERCACHE_TRANSIENT bool
: Is the inner write-through transient

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

MEMTYPE enum
: Memory type.

NSDESC enum
: Is secure side supposed to access secure or nonsecure memory

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TRANSIENT bool
: Is the outer write-through transient

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

PADDR unsigned int
: Physical address of the access.

PAGESIZE unsigned int
: Page size as log2(size).

SH enum
: Shareability

SIDE enum
: Inst / Data.

STAGE1_PERM unsigned int
: Stage 1 permissions, [5:3] Privileged access (XWR) [2:0] User access (XWR)

STAGE2_PERM unsigned int
: Stage 2 permission mask [5:3] Privileged access (XWR) [2:0] User access (XWR)

VADDR unsigned int
: Virtual address of the access.

VMID unsigned int
: Virtual machine identifier.

XS bool
: XS Attribute

nG enum
: Flag indicating whether ASID will be matched

MMU_TTB_READ

This event is triggered by reads caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data read.

IPA unsigned int
: For stage 1, the IPA of the read.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

OUTADDR unsigned int
: The output physical address.

PADDR unsigned int
: The physical address of the read.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MMU_TTB_WRITE

This event is triggered by writes caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data written.

IPA unsigned int
: For stage 1, the IPA of the write.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

PADDR unsigned int
: The physical address of the write.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MODE_CHANGE

Mode change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: The new mode.

NON_SECURE enum
: The new security state

OLD_MODE enum
: The old mode.

NOT_TAG_CHECKED_INST

MTE Tag not checked for current instruction.

PERIODIC

Called for every quantum. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

PC unsigned int
: The address of the next instruction to be executed on this CPU.

PMU_COUNTER_OVERFLOW

PMU counter overflow. Fields:

EVENT_ID unsigned int
: Event Identifier

INDEX unsigned int
: Counter index (as selected by PMSELR).

INTERRUPT bool
: Is interrupt enabled on overflow for this counter

PREFETCH_MEMORY64

Prefetch from PRFM or PRFUM instructions. Fields:

PRFOP unsigned int
: Prefetch hint

VADDR unsigned int
: Virtual address of the location that should be prefetched.

PRELOAD_DATA

Data preload from PLD instruction. Fields:

VADDR unsigned int
: Virtual address of the data that should be preloaded.

PRELOAD_INST

Instruction preload from PLI instruction. Fields:

VADDR unsigned int
: Virtual address of the instruction that should be preloaded.

PRE_CORE_LOAD

Trace just before a core load. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_CORE_STORE

Trace just before a core store. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_TTB_READ

This event is triggered before reads caused by a translation table walk. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR unsigned int
: The physical address of the read.

PSTATE_PACTIVE_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

PSTATE_PACTIVE_OPMODE_TRANSITION_DENIED

Trace the denied pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_POWER_TRANSITION

Trace pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

PSTATE_PACTIVE_POWER_TRANSITION_DENIED

Trace the denied pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_THREAD_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions for threads on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

RUN_STATE

Run state transition. Fields:

INST_COUNT unsigned int
: Ticks count at point of transition.

NEW enum
: New run state.

OLD enum
: Old run state.

SEMIHOSTING_CALL

Call of a semihost function occurred. Fields:

PC unsigned int
: The program counter after the semihosting call.

SEMIHOSTING_PRECALL

About to call semihost. Fields:

EL2 bool
: Use EL2 translation regime.

NS bool
: Is Non-Secure.

REG_WIDTH unsigned int
: The current register width in bytes.

SIGNAL

External signal state change. Fields:

SIGNAL enum
: Signal that changed

STATE bool
: Signal asserted state

SOFTWARE_STEP

Return the Debug Software Step state. Fields:

STATE enum
: Software Step state

SPSR

SPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: Which of the banked SPSR registers is written.

OLD_VALUE unsigned int
: The old SPSR value

VALUE unsigned int
: The new SPSR value

START_COMPILE

Compilation started. Fields:

VADDR unsigned int
: Instruction where compilation begins.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

STG_MULTIPLE_MICRO_OPS_ADDRESS_RANGE

Address range for MTE stores with multiple uops. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

STORE_MULTIPLE_REGS_ADDRESS_RANGE

Address range for stores with multiple registers. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

SUBOPTIMAL_LDST_RETIRED

suboptimal load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1: half-word, 2: word ...)

OPTIMAL_ALIGN_SELECTOR unsigned int
: Selector to decide the optimal alignment size (1: access width, 2: 4-bytes, 3: 8-bytes, ..., 12: 4-kbytes)

VADDR unsigned int
: The virtual address of the access

SYNC

Called for every synchronization, that is every quantum break. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

LOCAL_QUANTUM unsigned int
: The local quantum of this CPU.

LOCAL_TIME unsigned int
: The local time of this CPU.

SYNC_REASON enum
: Reason for sync.

SYSCALL

System call instruction executed. Fields:

IMM unsigned int
: Immediate value of the system call instruction.

TYPE enum
: System call type.

VADDR unsigned int
: Instruction that caused the system call.

SYSREG_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE32

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE64

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

UNALIGNED_LDST_RETIRED

Processor unaligned load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1=>half-word, 2->word ...)

NUMBER_OF_BEATS unsigned int
: Number of accesses (beats) in this burst (Total size in bytes is ACCESS_SIZE * NUMBER_OF_BEATS)

VADDR unsigned int
: The virtual address of the access

VFP_D_REGS

VFP/NEON D 64 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_Q_REGS

VFP/NEON Q 128 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding byte in the register. For example, 0x1 would indicate a write to VALUE[7:0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_SYS_REGS

Writes to the VFP/NEON units system registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: Which VFP system register is written.

MASK unsigned int
: The mask indicating updated cumulative bits.

OLD_VALUE unsigned int
: The register's old value overwritten.

VALUE unsigned int
: The new value written to the VFP system register.

VFP_S_REGS

VFP/NEON S 32 bit register write. Fields:

ALIASING enum
: Type of register aliasing used.

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VALUE unsigned int
: The new value written to the register.

WAYPOINT

Signals end of basic block execution. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: Origin instructions set

IS_COND bool
: Indicates if this is a conditional waypoint.

PC unsigned int
: Origin address (or 0 if unavailable)

TAKEN bool
: Indicates if this waypoint was taken.

TARGET unsigned int
: Destination address

TARGET_ISET enum
: Destination instructions set

WFE_END

WFE ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFE.

WFE_EVENT_REGISTER

WFE event register status: set/clear, reason. Fields:

INST_COUNT unsigned int
: Ticks count.

REASON enum
: Reason for set/clear. Only REASON==0 (Cleared by WFE) clears the bit, all other reasons set it.

WFE_IGNORED

WFE ignored. Fields:

EVENT bool
: This WFE was ignored because the event register was set.

INST_COUNT unsigned int
: Ticks count when ignoring WFE.

TRAPPED bool
: This WFE was trapped.

WFE_START

WFE entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFE.

WFI_END

WFI ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFI.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

WFI_IGNORED

WFI ignored. Fields:

DISABLED bool
: This WFI was ignored because WFI is disabled.

INST_COUNT unsigned int
: Ticks count when ignoring WFI.

REASON enum
: specifies reason why WFI trace was ignored

TRAPPED bool
: This WFI was trapped.

WFI_START

WFI entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFI.

WFI_WAKEUP

WFI wakeup. Fields:

INST_COUNT unsigned int
: Ticks count when WFI wakeup occurred.

REASON enum
: Reason for wakeup.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.17 ARM_Cortex-A7 - trace

This section describes the trace sources.

ASYNC_MEMORY_FAULT

Context ID Register write. Fields:

FAULT unsigned int
: Fault status in ESR format

PADDR unsigned int
: Physical Address (or 0 if unavailable)

VADDR signed int
: Virtual Address (or 0 if unavailable)

ATOMIC_END_ACCESS

Bus trace access for atomics. Fields:

ACCESS_FAIL bool
: Memory access failed

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

LOAD_VALUE unsigned int
: Loaded values

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ATOMIC_START_ACCESS

Bus trace access for atomics. Fields:

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

NSDESC unsigned int
: The security state of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ArchMsg.Error.exit_code

Fields:

EXITCODE signed int
: Exit code to be returned

KIND string
: Component kind that invoked the exit code trace

REASON string
: Reason why the exit code trace was invoked (optional)

ArchMsg.Warning.IgnoredWarmResetRequest

Event fires when software attempts to request warm reset through write to deprecated DBGPRCR.CWRR.

ArchMsg.Warning.TarmacReconstructor

Unexpected event happened during Tarmac reconstruction. Fields:

REASON string
: Reason

ArchMsg.Warning.Unpredictably Indexed PM Event Register

Fields:

IsDirect unsigned int
: Direct Access rather than indirect

N unsigned int
: Number of Accessible Registers

SEL unsigned int
: Selector

ArchMsg.Warning.branch_to_unaligned_address

Fields:

ADDR unsigned int
: Unaligned branch address

ArchMsg.Warning.cache_contents_unknown

A cache has been activated without being invalidated and may contain UNKNOWN content. Fields:

REGIME unsigned int
: Current translation regime

SIDE unsigned int
: 0 d-side cache or unified cache, 1 i-side

ArchMsg.Warning.dcimvac_matches_watchpoint

Watchpoints matching an AArch32 DCIMVA are implementation defined.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_fieldmismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_fracbitsnegative

VCVT Instruction is unpredictable with a negative number of fraction bits.

ArchMsg.Warning.decode_initblock

Instruction is unpredictable in an IT block.

ArchMsg.Warning.decode_initblocknotlast

Instruction is unpredictable in an IT block when not the last.

ArchMsg.Warning.decode_invalid_condition

Instruction is unpredictable when the condition is not AL.

ArchMsg.Warning.decode_invalidfieldcombination

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_invalidvalue

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_mem_hint_unallocated

There is no memory hint allocated to this bit pattern.

ArchMsg.Warning.decode_registermatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registermismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registeroutofrange

Float point instruction is unpredictable indexing beyond the end of the register bank.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_sbzsbo

Reserved bits in the instruction are not canonical.

ArchMsg.Warning.decode_transactiontoobig

Attempt to perform a memory transaction of more than 128 bytes.

ArchMsg.Warning.decode_unpred_other

Instruction is unpredictable.

ArchMsg.Warning.decode_unpreduseofpc

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13inreglist

Instruction is unpredictable with R13 in register-list.

ArchMsg.Warning.decode_useofd16orhigher

Attempt to use register D16 or higher where it is not allowed.

ArchMsg.Warning.decode_writebackandbaseinlist

Instruction is unpredictable with write back when the base register is in the transfer list.

ArchMsg.Warning.decode_zeroregistersinlist

Instruction is unpredictable when no registers are to be transfered.

ArchMsg.Warning.recursive_branch

An instruction is performing a branch that targets the same instruction. If this was intended then a WFI might be more effective. Fields:

PC unsigned int
: Address of instruction causing the branch

ArchMsg.Warning.recursive_exception

An instruction has generated an exception that targets the same instruction. Fields:

CPSR unsigned int
: Processor state

PC unsigned int
: Address of instruction causing the exception

TYPE enum
: Type of exception

ArchMsg.Warning.reentrant_vector_catch

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR unsigned int
vector address

ArchMsg.Warning.reserved_it_state

Fields:

ITSTATE unsigned int
reserved IT state

ArchMsg.Warning.tlb_contents_unknown

Translation has been enabled without invalidating the TLB. Fields:

EL unsigned int
Exception level of translation regime
INVALIDITY unsigned int
Which TLB may be invalid (if TLBs are separate)
NON_SECURE unsigned int
Security state of translation regime
VMID unsigned int
Current VMID, if applicable

ArchMsg.Warning.unpredictable_a32_breakpoint

Fields:

ADDR1 unsigned int
Address
ADDR2 unsigned int
Breakpoint address
BAS unsigned int
BAS Field
IS_ADDRESS_MISMATCH_BREAKPOINT bool
Is Mismatch breakpoint, not a match breakpoint
N unsigned int
Breakpoint number
OFFSET unsigned int
Indicates which halfword actually considering into the A32 instruction

ArchMsg.Warning.unpredictable_banked_register_access

Fields:

R unsigned int
R bit
SYSM unsigned int
SYSm

ArchMsg.Warning.unpredictable_ccfail_undef

Fields:

OPCODE unsigned int
opcode of the instruction
VADDR unsigned int
Virtual address of the instruction being executed

ArchMsg.Warning.unpredictable_ici_bits

ICI bits do not apply or are architecturally unpredictable for instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.unpredictable_t32_breakpoint

Fields:

ADDR1 unsigned int
: Address

ADDR2 unsigned int
: Breakpoint address

BAS unsigned int
: BAS Field

IS_ADDRESS_MISMATCH_BREAKPOINT bool
: Is Mismatch breakpoint, not a match breakpoint

N unsigned int
: Breakpoint number

OFFSET unsigned int
: Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_t32_vector_catch

Fields:

ADDR unsigned int
: Address

OFFSET unsigned int
: Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_watchpoint_far

Fields:

LOWER_BOUND unsigned int
: the lowest address accessed by the instruction that triggered the watchpoint

UPPER_BOUND unsigned int
: the highest watchpointed address accessed by that instruction

ArchMsg.Warning.warning_AdvSIMDExpandImmUnexpectedZero

AdvSIMDExpandImm may treat this immediate value as UNPREDICTABLE.

ArchMsg.Warning.warning_ConditionalSMC

SMC instruction has UNPREDICTABLE effects when conditional when combined with traps.

ArchMsg.Warning.warning_access_crosses_page_boundary

Access crosses page boundary.

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault_on_both_pages

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_spanning_different_memory

Fields:

ADDR unsigned int
: address of access crossing page

MEMTYPE_PAGE1 signed int
: First Page's memory type (0-Normal, 1-Non-Normal)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MEMTYPE_PAGE2 signed int
Second Page's memory type (0-Normal, 1-Non-Normal)

ArchMsg.Warning.warning_access_wraps_around_memory

Fields:

WIDTH unsigned int
Address width

ArchMsg.Warning.warning_bx_from_thumbee

Fields:

ADDR unsigned int
Destination address

ArchMsg.Warning.warning_ccsidr_unimplemented_level

Fields:

CSSELR unsigned int
current effective value of CSSELR

ArchMsg.Warning.warning_change_to_ns_when_tge_set

Attempting to change NS to 1 when HCR.TGE = 1.

ArchMsg.Warning.warning_contiguous_bit_check_abort

Fields:

ENTRY_ADDR unsigned int
address of conflicting TLB entry

ArchMsg.Warning.warning_contiguous_bit_error

Fields:

CONTIG_ADDR unsigned int
address of the TLB first read, with which this entry is expected to be contiguous
CONTIG_DATA unsigned int
expected contents based on the entry at CONTIG_ADDR
ENTRY_ADDR unsigned int
address of conflicting TLB entry
ENTRY_DATA unsigned int
contents of conflicting TLB entry

ArchMsg.Warning.warning_cp10_cp11_mismatch

Fields:

EL unsigned int
EL owning the control

ArchMsg.Warning.warning_cp10_cp11_reserved_value

Fields:

CP10 unsigned int
CP10 access
CP11 unsigned int
CP11 access

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_csselr_level_out_of_range

Fields:

IMPLEMENTED unsigned int
: number of cache levels implemented

LEVEL unsigned int
: written value of CSSELR.Level

TYPE enum
: The type of cache selected by CSSELR.InD and TnD

ArchMsg.Warning.warning_debug_flow_control_bits_not_obeyed

Fields:

ERROR enum
: type of error resulting

REGISTER enum
: register under consideration

ArchMsg.Warning.warning_debug_register_access_during_reset

Fields:

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

ArchMsg.Warning.warning_decode_cps_inconsistent_fields

The mode setting fields in the CPS instruction are inconsistent.

ArchMsg.Warning.warning_decode_invalid_state

Instruction is unpredictable in current exception-level/security state.

ArchMsg.Warning.warning_default_cacheable_mmu_on

Default cacheable is enabled (HCR.DC = 1) while MMU is enabled.

ArchMsg.Warning.warning_default_cacheable_vmmu_off

Default cacheable is enabled (HCR.DC = 1) while HCR.VM = 0.

ArchMsg.Warning.warning_deprecated_wvr_bit_2_set

Fields:

N unsigned int
: index

WVR unsigned int
: DBGWVR

ArchMsg.Warning.warning_eret_while_software_step_active_pending

Missing ISB between setting MDSCR_EL1.SS and ERET.

ArchMsg.Warning.warning_exclusive_to_non_normal

Fields:

ADDR unsigned int
: Address

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_exclusive_to_non_writeback

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_execute_from_device_memory

An attempt was made to execute from Device memory.

ArchMsg.Warning.warning_illegal_cpsr_mode

Fields:

NEW_MODE unsigned int
: New value of CPSR.M

ArchMsg.Warning.warning_illegal_srs_mode

Fields:

MODE unsigned int
: mode for Banked SP

ArchMsg.Warning.warning_implementation_defined_read_debug_register_in_SCS

Read Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_implementation_defined_sequential_security_transitions_supported

The behaviour of a sequential instruction fetches that cross from non-secure to secure memory and contain SG instruction is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_implementation_defined_stack_limit_check_supported

It is IMPLEMENTATION DEFINED whether stack pointer limit checking is performed for the this instructions. Fields:

ADDRESS unsigned int
: Address of instruction

OPCODE unsigned int
: instruction opcode

ArchMsg.Warning.warning_implementation_defined_write_debug_register_in_SCS

Write Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

DATA unsigned int
: data attempted to be written

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_invalid_tcr_granule

Fields:

REQUEST signed int
: page size requested (or 0 for reserved)

SUBSTITUTE signed int
: best guess available page size

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

TG_ID bool
bits TG0 or TG1

ArchMsg.Warning.warning_load_multiple_user_registers_from_user_mode

An LDM(user registers) instruction executed from user mode.

ArchMsg.Warning.warning_load_pc_from_unaligned

PC loaded from an unaligned location.

ArchMsg.Warning.warning_mte_in_mem_attr_not_supported

Fields:

ATTR unsigned int
Memory attribute

ArchMsg.Warning.warning_reserved_breakpoint_state_match

Fields:

IS_BREAKPOINT unsigned int
Is breakpoint, not watchpoint

ArchMsg.Warning.warning_shareability

Unpredictable: combination of the 1st and 2nd stages of translation is Normal Inner Non-Cacheable, Outer Non-Cacheable.

ArchMsg.Warning.warning_software_step_set_while_enabled

MDSCR_EL1.SS set to 1 while software step debug exceptions are enabled.

ArchMsg.Warning.warning_thumb_instruction_wraps_around_memory

Thumb instruction warps around memory.

ArchMsg.Warning.warning_ttbr_sbz_bits_are_not_zero

Fields:

MSB unsigned int
x-1
TTBR enum
which TTBR register

ArchMsg.Warning.warning_unaligned_address_dbgdtrrx_write

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_address_dbgdtrtx_read

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_device

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_strongly_ordered

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unknown_sau_rnr

SAU_RNR was set to an unsupported value.

ArchMsg.Warning.warning_unpredictable_AIRCR_PRIP_and_BFHFNMINP

The effect of setting both AIRCR.BFHFNMINP and AIRCR.PRIP to 1 is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTCLRACTIVE_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTCLRACTIVE if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_and_SYSRESETREQ

When the processor is halted in Debug state, if a write to the register writes a 1 to both VECTRESET and SYSRESETREQ, the behavior is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTRESET if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_incorrect_VKEY

The value 0x05FA must be written to AIRCR.VKEY, otherwise the register write is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_EXE_RETURN_Reserved_Bit

EXC_RETURN[23:7] are reserved with the special condition that all bits should be written as one. Values other than all 1s are UNPREDICTABLE. Fields:

EXC_RETURN unsigned int
Special PC value
SBO_MASK unsigned int
Which bits Should Be One to not UNPRED

ArchMsg.Warning.warning_unpredictable_change_to_DEMCR_MON_STEP_at_insufficient_priority

The effect of changing DEMCR.MON_STEP at an execution priority that is lower than the priority of the DebugMonitor exception is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_change_to_priority_of_active_exception

Changing the priority of an active exception is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_clear_lspact

Fields:

DRVEXC_TAKEN unsigned int
derived exception taken not pended

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

REACHED_LAST_STORE unsigned int
only the last faulted

ArchMsg.Warning.warning_unpredictable_dbg_exit_unaligned_dlr

Debug exit to AArch32 with DLR[0] set to 1. PC[0] can be set to 0 or to DLR[0].

ArchMsg.Warning.warning_unpredictable_exception_catch

Generation of an exception catch event is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_inconsistent_state

state on exception return is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_instruction

exception return instruction is unpredictable.

ArchMsg.Warning.warning_unpredictable_in_debug_state

Instruction is UNPREDICTABLE when executed in debug state.

ArchMsg.Warning.warning_unpredictable_mair_encoding

Fields:

ATTR unsigned int
Memory attribute

ArchMsg.Warning.warning_unpredictable_pmu_counter_access

Access to PMU counters from non-secure EL0 or EL1 is UNPREDICTABLE with MDCR_EL2.HPMN set to 0.

ArchMsg.Warning.warning_unpredictable_prioritization_breakpoint_match_vector_catch

Two events with the same priority occurred at the same instruction: Address Matching Vector Catch debug event, and Breakpoint debug event It is constrained unpredictable which is taken.

ArchMsg.Warning.warning_unpredictable_stack_selection

UNPREDICTABLE state on context switch.

ArchMsg.Warning.warning_unpredictable_tsize_out_of_range

Fields:

TCR enum
Which TCR
TSIZE signed int
Value of TSize
TSIZE_MAX signed int
The maximum allowed value
TTBR unsigned int
Which TTBR region

ArchMsg.Warning.warning_unpredictable_unaligned_pc_as_base_register

The PC value must be word-aligned, otherwise the behavior of the instruction is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_unaligned_pop_stack

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

FRAME_ADDR unsigned int
: unaligned stack frame address

ArchMsg.Warning.warning_unpredictable_vmsa_memattrib

Fields:

TEXCB unsigned int
: texcb value

TEX_REMAP bool
: use tex remap

ArchMsg.Warning.warning_unpredictable_vtcr_t0sz_sl0

Fields:

SL0 signed int
: VTCR.SL0, starting level for stage 2 translation table walks

T0SZ signed int
: VTCR.T0SZ, required input address range

ArchMsg.Warning.warning_unpredictable_write_DHCSR

Fields:

NEW unsigned int
: data attempted to be written

OLD unsigned int
: previous value

ArchMsg.Warning.warning_unpredictable_write_SHCSR

UNPREDICTABLE setting a pending bit to 1 for an exception with priority >= execution priority.

ArchMsg.Warning.warning_unsupported_access_to_memory_mapped_register

Access to memory mapped register is unsupported.

ArchMsg.Warning.warning_user_jmcr_access

Fields:

WRITE unsigned int
: access is write

ArchMsg.Warning.warning_wcr_mask_and_bas

DBGWCR_EL1.MASK is non-zero and BAS is not 0xff.

ArchMsg.Warning.warning_wcr_mask_reserved

DBGWCR_ELn.MASK is set to a reserved value.

ArchMsg.Warning.warning_wcr_non_configuous_bas

DBGWCR_EL1.BAS has non-contiguous set of ones.

ArchMsg.Warning.warning_wvr_masked_bit_not_zero

DBGWVRn_EL1.VA contains a bit masked by DBGWCRn_EL1.MASK that is not zero.

BRANCH_MISPREDICT

Simulating branch mispredict. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PC unsigned int
: Origin address (or 0 if unavailable)

BRA_DIR

Direct branches, to immediate address. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR

Indirect branches, perhaps to a register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR_CCFAIL

Indirect branches fail, perhaps to a register.

BRA_RET_CCFAIL

Branche return fail, perhaps to a register.

CACHE_MAINTENANCE_OP

Cache Maintenance Operation. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Specified MVA or Set/way

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

FUNCTION enum
: Clean / Invalidate

SCOPE enum
: Affected region

SIDE enum
: Inst / Data.

CCFAIL

Conditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCFAIL_UNC

Unconditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS

Conditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS_UNC

Unconditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CODE_CACHE_MAINT

Code cache maintenance. Fields:

END_INCL unsigned int
: Inclusive end addres for by VA or PA requests

NS bool
: Non-secure world for PA

START unsigned int
: Start address for by VA or PA requests

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

TYPE enum
Request type.

COMPILE_BLOCK_END

Last instruction of basic block translated. Fields:

VADDR unsigned int
Address of next instruction after this basic block

COMPILE_BLOCK_START

First instruction of basic block translated. Fields:

VADDR unsigned int
Address of first instruction in basic block

COMPILE_INST

ARM instruction compiled. Fields:

DISASS string
Disassembly of the instruction.
ISET enum
The instruction set of this instruction.
ITSTATE unsigned int
The ITSTATE current for the instruction.
OPCODE unsigned int
The opcode of the instruction.
PC unsigned int
The address of the instruction.
SIZE unsigned int
The size of the instruction in bytes.

CONTEXTIDR

Context ID Register write. Fields:

CORE_NUM unsigned int
Core number in a multi processor.
NS bool
Secure or nonsecure banked register is accessed.
UNDEF bool
The register accessed is undefined.
VALUE unsigned int
The new value written.

CONTEXT_SYNC

Called for every context synchronization event. Fields:

ARCH_CSE bool
Architectural Context Synchronization Event
SYSREG_SYNC bool
System register synchronization event

CORE_ENDIAN

Core BE8 Big-Endian state changed. Fields:

BE8 bool
Core BE8 Big-Endian state

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_INFO

Static processor attributes. Only triggered by a call to DumpState(). Fields:

ARCH_PROFILE enum
: The architecture profile of the core.

CLUSTER_ID unsigned int
: The cluster ID of this processor.

CORE_NUM unsigned int
: The number of this core in an MP processor.

DETAILED_CORE_FEATURES enum
: Detailed core features as a bitfield. Enum values are bitmasks to extract the info from the field

FPU_VERSION enum
: The VFP version implemented by the core.

MEM_ARCH enum
: The memory architecture of the core.

MIDR_EL1 unsigned int
: MIDR_EL1 value

NUM_CORES unsigned int
: The number of cores in this MP processor.

QUANTUM_SIZE unsigned int
: The default quantum size of the core.

SECURITY_FEATURES bool
: Does the core have security features?

CORE_LOADS

Processor load accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CORE_REGS

Changes of the core registers R0 to R14. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number, 0 to 14.

MODE enum
: Bank of the register accessed.

OLD_VALUE unsigned int
: The old value overwritten.

PHYS_ID enum
: The physical register accessed.

VALUE unsigned int
: The new value written to the register.

CORE_STORES

Processor store accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CP14_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP14_WRITE

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

CP15_WRITE

System Control Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

CPSR

CPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

OLD_VALUE unsigned int
: The old CPSR value

UNKNOWN unsigned int
: Bits within the register that have unknown value.

VALUE unsigned int
: The new CPSR value

CRYPTO_SPEC

Every crypto instruction speculatively executed.

CorePowerStateChange

Core power-up state has changed. Fields:

PU bool
: Whether the core powered up

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DATA_CACHE_SET_ALLOC_TAGS

Set allocation tags from DC GVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of memory in bytes.

TAG unsigned int
: The value of the tag being set.

VADDR unsigned int
: Virtual address used for setting allocation tags.

DATA_CACHE_ZERO

Zero and invalidate cache line from DC ZVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of zero'd memory in bytes.

VADDR unsigned int
: Virtual address of the zero'd cache line.

DEBUG_EVENT

Hardware debug support event. Fields:

EVENT enum
: Description of event

VALUE unsigned int
: data value

END_COMPILE

Compilation end. Fields:

END_OF_PAGE_COUNT unsigned int
: Number of basic blocks exited due to page boundary since START_COMPILE.

FETCHFAIL_COUNT unsigned int
: Number of basic blocks exited due to ifetch failure START_COMPILE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: Number of instructions compiled since START_COMPILE.

NONSEQ_COUNT unsigned int
: Number of basic blocks exited due to non-sequential instructions since START_COMPILE.

PAGE_STRADDLE_COUNT unsigned int
: Number of basic blocks exited due to unaligned instructions crossing page since START_COMPILE.

EXCEPTION

Exception taken. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

IS_PHYSICAL bool
: Physical or Virtual exception

LR unsigned int
: The value assigned to the link register.

PC unsigned int
: The location where the exception occurred.

PREFERRED_RETURN unsigned int
: The preferred return address for the exception.

TARGET_ISET enum
: The instruction set of the exception handler code.

TARGET_PC unsigned int
: The address the exception branches to.

VECTOR enum
: The exception vector.

VECTOR_OFFSET unsigned int
: The offset of an exception vector from the base address.

EXCEPTION_END

Every exception completed.

EXCEPTION_RAISE

Every exception raised.

EXCEPTION_RETURN

Branches on leaving exception. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

EXCEPTION_RETURN_PREBRANCH

Exception return event that must occur before PC and processor state have updated.

EXCEPTION_START

Every exception started.

FP_STATE

Floating point state. Fields:

FPSR unsigned int
: FPSR register value.

MASK unsigned int
: The mask indicating updated cumulative bits.

HLT

HLT instruction occurred. Fields:

IMM16 unsigned int
: The 16-bit immediate value encoded in HLT instruction.

INST

Every instruction executed. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DEBUG_STATE bool
: The instruction is executed in debug state.

DISASS string
: Disassembly of instruction.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The current instruction set.

ITSTATE unsigned int
: The current ITSTATE.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

MODE enum
: The mode the core is in.

NS unsigned int
: The core's non-secure bit.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

OPCODE unsigned int
: The opcode of the instruction.

PADDR unsigned int
: The physical address of the instruction.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the instruction.

PC unsigned int
: The address of the instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SIZE unsigned int
: The size of the instruction in bytes.

INST_END

Every instruction completed.

INST_START

Every instruction started. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

INST_STRADDLE

Instruction straddles boundary. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

ISB

ISB instruction occurred.

LOAD_MULTIPLE_REGS_LIST

List of destination registers (not source) of a multiple registers load instruction. Fields:

REG string
: Register to add into the list

LOCAL_MONITOR

Local monitor activity. Fields:

PADDR unsigned int
: Local Monitor Address

State enum
: State of the monitor (Open/Exclusive)

MEMMAP_DEBUG_READ

Memory mapped reads to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

MEMMAP_DEBUG_WRITE

Memory mapped writes to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

MMU_TRANS

Address translation information. Fields:

ASID unsigned int
: Address space identifier.

CORE_NUM unsigned int
: Core number in a multi processor.

Hyp bool
: Entry matches in Hyp state only

INNERCACHE_RA bool
: Is the inner cache allocate on read

INNERCACHE_TRANSIENT bool
: Is the inner write-through transient

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

MEMTYPE enum
: Memory type.

NSDESC enum
: Is secure side supposed to access secure or nonsecure memory

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TRANSIENT bool
: Is the outer write-through transient

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PADDR unsigned int
: Physical address of the access.

PAGESIZE unsigned int
: Page size as log2(size).

SH enum
: Shareability

SIDE enum
: Inst / Data.

STAGE1_PERM unsigned int
: Stage 1 permissions, [5:3] Privileged access (XWR) [2:0] User access (XWR)

STAGE2_PERM unsigned int
: Stage 2 permission mask [5:3] Privileged access (XWR) [2:0] User access (XWR)

VADDR unsigned int
: Virtual address of the access.

VMID unsigned int
: Virtual machine identifier.

XS bool
: XS Attribute

nG enum
: Flag indicating whether ASID will be matched

MMU_TTB_READ

This event is triggered by reads caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data read.

IPA unsigned int
: For stage 1, the IPA of the read.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

OUTADDR unsigned int
: The output physical address.

PADDR unsigned int
: The physical address of the read.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MMU_TTB_WRITE

This event is triggered by writes caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data written.

IPA unsigned int
: For stage 1, the IPA of the write.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

PADDR unsigned int
: The physical address of the write.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MODE_CHANGE

Mode change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: The new mode.

NON_SECURE enum
: The new security state

OLD_MODE enum
: The old mode.

NOT_TAG_CHECKED_INST

MTE Tag not checked for current instruction.

PERIODIC

Called for every quantum. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

PC unsigned int
: The address of the next instruction to be executed on this CPU.

PMU_COUNTER_OVERFLOW

PMU counter overflow. Fields:

EVENT_ID unsigned int
: Event Identifier

INDEX unsigned int
: Counter index (as selected by PMSELR).

INTERRUPT bool
: Is interrupt enabled on overflow for this counter

PRELOAD_DATA

Data preload from PLD instruction. Fields:

VADDR unsigned int
: Virtual address of the data that should be preloaded.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PRELOAD_INST

Instruction preload from PLI instruction. Fields:

VADDR unsigned int
: Virtual address of the instruction that should be preloaded.

PRE_CORE_LOAD

Trace just before a core load. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_CORE_STORE

Trace just before a core store. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_TTB_READ

This event is triggered before reads caused by a translation table walk. Fields:

ADDR unsigned int
: The physical address of the read.

PSTATE_PACTIVE_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

PSTATE_PACTIVE_OPMODE_TRANSITION_DENIED

Trace the denied pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_POWER_TRANSITION

Trace pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OLD_STATE enum
: Power mode before the transition

PSTATE_PACTIVE_POWER_TRANSITION_DENIED

Trace the denied pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_THREAD_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions for threads on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

RUN_STATE

Run state transition. Fields:

INST_COUNT unsigned int
: Ticks count at point of transition.

NEW enum
: New run state.

OLD enum
: Old run state.

SEMIHOSTING_CALL

Call of a semihost function occurred. Fields:

PC unsigned int
: The program counter after the semihosting call.

SEMIHOSTING_PRECALL

About to call semihost. Fields:

EL2 bool
: Use EL2 translation regime.

NS bool
: Is Non-Secure.

REG_WIDTH unsigned int
: The current register width in bytes.

SIGNAL

External signal state change. Fields:

SIGNAL enum
: Signal that changed

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

STATE bool
: Signal asserted state

SPSR

SPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: Which of the banked SPSR registers is written.

OLD_VALUE unsigned int
: The old SPSR value

VALUE unsigned int
: The new SPSR value

START_COMPILE

Compilation started. Fields:

VADDR unsigned int
: Instruction where compilation begins.

STG_MULTIPLE_MICRO_OPS_ADDRESS_RANGE

Address range for MTE stores with multiple uops. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

STORE_MULTIPLE_REGS_ADDRESS_RANGE

Address range for stores with multiple registers. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

SUBOPTIMAL_LDST_RETIRED

suboptimal load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1: half-word, 2: word ...)

OPTIMAL_ALIGN_SELECTOR unsigned int
: Selector to decide the optimal alignment size (1: access width, 2: 4-bytes, 3: 8-bytes, ..., 12: 4-kbytes)

VADDR unsigned int
: The virtual address of the access

SYNC

Called for every synchronization, that is every quantum break. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

LOCAL_QUANTUM unsigned int
: The local quantum of this CPU.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOCAL_TIME unsigned int
: The local time of this CPU.

SYNC_REASON enum
: Reason for sync.

SYSCALL

System call instruction executed. Fields:

IMM unsigned int
: Immediate value of the system call instruction.

TYPE enum
: System call type.

VADDR unsigned int
: Instruction that caused the system call.

SYSREG_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE32

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE64

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

UNALIGNED_LDST_RETIRED

Processor unaligned load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1=>half-word, 2->word ...)

NUMBER_OF_BEATS unsigned int
: Number of accesses (beats) in this burst (Total size in bytes is ACCESS_SIZE * NUMBER_OF_BEATS)

VADDR unsigned int
: The virtual address of the access

VFP_D_REGS

VFP/NEON D 64 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_Q_REGS

VFP/NEON Q 128 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding byte in the register. For example, 0x1 would indicate a write to VALUE[7:0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_SYS_REGS

Writes to the VFP/NEON units system registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: Which VFP system register is written.

MASK unsigned int
: The mask indicating updated cumulative bits.

OLD_VALUE unsigned int
: The register's old value overwritten.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VALUE unsigned int
: The new value written to the VFP system register.

VFP_S_REGS

VFP/NEON S 32 bit register write. Fields:

ALIASING enum
: Type of register aliasing used.

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

WAYPOINT

Signals end of basic block execution. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: Origin instructions set

IS_COND bool
: Indicates if this is a conditional waypoint.

PC unsigned int
: Origin address (or 0 if unavailable)

TAKEN bool
: Indicates if this waypoint was taken.

TARGET unsigned int
: Destination address

TARGET_ISET enum
: Destination instructions set

WFE_END

WFE ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFE.

WFE_EVENT_REGISTER

WFE event register status: set/clear, reason. Fields:

INST_COUNT unsigned int
: Ticks count.

REASON enum
: Reason for set/clear. Only REASON==0 (Cleared by WFE) clears the bit, all other reasons set it.

WFE_IGNORED

WFE ignored. Fields:

EVENT bool
: This WFE was ignored because the event register was set.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: Ticks count when ignoring WFE.

TRAPPED bool
: This WFE was trapped.

WFE_START

WFE entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFE.

WFI_END

WFI ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFI.

WFI_IGNORED

WFI ignored. Fields:

DISABLED bool
: This WFI was ignored because WFI is disabled.

INST_COUNT unsigned int
: Ticks count when ignoring WFI.

REASON enum
: specifies reason why WFI trace was ignored

TRAPPED bool
: This WFI was trapped.

WFI_START

WFI entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFI.

WFI_WAKEUP

WFI wakeup. Fields:

INST_COUNT unsigned int
: Ticks count when WFI wakeup occurred.

REASON enum
: Reason for wakeup.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.18 ARM_Cortex-A72 - trace

This section describes the trace sources.

ASYNC_MEMORY_FAULT

Context ID Register write. Fields:

FAULT unsigned int
: Fault status in ESR format

PADDR unsigned int
: Physical Address (or 0 if unavailable)

VADDR signed int
: Virtual Address (or 0 if unavailable)

ATOMIC_END_ACCESS

Bus trace access for atomics. Fields:

ACCESS_FAIL bool
: Memory access failed

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

LOAD_VALUE unsigned int
: Loaded values

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ATOMIC_START_ACCESS

Bus trace access for atomics. Fields:

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

NSDESC unsigned int
: The security state of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ArchMsg.Error.exit_code

Fields:

EXITCODE signed int
: Exit code to be returned

KIND string
: Component kind that invoked the exit code trace

REASON string
: Reason why the exit code trace was invoked (optional)

ArchMsg.Warning.IgnoredWarmResetRequest

Event fires when software attempts to request warm reset through write to deprecated DBGPRCR.CWRR.

ArchMsg.Warning.TarmacReconstructor

Unexpected event happened during Tarmac reconstruction. Fields:

REASON string
: Reason

ArchMsg.Warning.Unpredictably Indexed PM Event Register

Fields:

IsDirect unsigned int
: Direct Access rather than indirect

N unsigned int
: Number of Accessible Registers

SEL unsigned int
: Selector

ArchMsg.Warning.branch_to_unaligned_address

Fields:

ADDR unsigned int
: Unaligned branch address

ArchMsg.Warning.cache_contents_unknown

A cache has been activated without being invalidated and may contain UNKNOWN content. Fields:

REGIME unsigned int
: Current translation regime

SIDE unsigned int
: 0 d-side cache or unified cache, 1 i-side

ArchMsg.Warning.dcimvac_matches_watchpoint

Watchpoints matching an AArch32 DCIMVA are implementation defined.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_fieldmismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_fracbitsnegative

VCVT Instruction is unpredictable with a negative number of fraction bits.

ArchMsg.Warning.decode_initblock

Instruction is unpredictable in an IT block.

ArchMsg.Warning.decode_initblocknotlast

Instruction is unpredictable in an IT block when not the last.

ArchMsg.Warning.decode_invalid_condition

Instruction is unpredictable when the condition is not AL.

ArchMsg.Warning.decode_invalidfieldcombination

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_invalidvalue

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_mem_hint_unallocated

There is no memory hint allocated to this bit pattern.

ArchMsg.Warning.decode_registermatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registermismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registeroutofrange

Float point instruction is unpredictable indexing beyond the end of the register bank.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_sbzsbo

Reserved bits in the instruction are not canonical.

ArchMsg.Warning.decode_transactiontoobig

Attempt to perform a memory transaction of more than 128 bytes.

ArchMsg.Warning.decode_unpred_other

Instruction is unpredictable.

ArchMsg.Warning.decode_unpreduseofpc

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13inreglist

Instruction is unpredictable with R13 in register-list.

ArchMsg.Warning.decode_useofd16orhigher

Attempt to use register D16 or higher where it is not allowed.

ArchMsg.Warning.decode_writebackandbaseinlist

Instruction is unpredictable with write back when the base register is in the transfer list.

ArchMsg.Warning.decode_zeroregistersinlist

Instruction is unpredictable when no registers are to be transfered.

ArchMsg.Warning.recursive_branch

An instruction is performing a branch that targets the same instruction. If this was intended then a WFI might be more effective. Fields:

PC unsigned int
: Address of instruction causing the branch

ArchMsg.Warning.recursive_exception

An instruction has generated an exception that targets the same instruction. Fields:

CPSR unsigned int
: Processor state

PC unsigned int
: Address of instruction causing the exception

TYPE enum
: Type of exception

ArchMsg.Warning.reentrant_vector_catch

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR unsigned int
vector address

ArchMsg.Warning.reserved_it_state

Fields:

ITSTATE unsigned int
reserved IT state

ArchMsg.Warning.tlb_contents_unknown

Translation has been enabled without invalidating the TLB. Fields:

EL unsigned int
Exception level of translation regime
INVALIDITY unsigned int
Which TLB may be invalid (if TLBs are separate)
NON_SECURE unsigned int
Security state of translation regime
VMID unsigned int
Current VMID, if applicable

ArchMsg.Warning.unknown_DLR_DSPSR

Exiting debug state with unknown DLR or DSPSR.

ArchMsg.Warning.unknown_ELR_SPSR

Returning from debug exception with unknown ELR or SPSR.

ArchMsg.Warning.unpredictable_a32_breakpoint

Fields:

ADDR1 unsigned int
Address
ADDR2 unsigned int
Breakpoint address
BAS unsigned int
BAS Field
IS_ADDRESS_MISMATCH_BREAKPOINT bool
Is Mismatch breakpoint, not a match breakpoint
N unsigned int
Breakpoint number
OFFSET unsigned int
Indicates which halfword actually considering into the A32 instruction

ArchMsg.Warning.unpredictable_banked_register_access

Fields:

R unsigned int
R bit
SYSM unsigned int
SYSm

ArchMsg.Warning.unpredictable_ccfail_undef

Fields:

OPCODE unsigned int
opcode of the instruction

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VADDR unsigned int
Virtual address of the instruction being executed

ArchMsg.Warning.unpredictable_ici_bits

ICI bits do not apply or are architecturally unpredictable for instruction.

ArchMsg.Warning.unpredictable_t32_breakpoint

Fields:

ADDR1 unsigned int
Address
ADDR2 unsigned int
Breakpoint address
BAS unsigned int
BAS Field
IS_ADDRESS_MISMATCH_BREAKPOINT bool
Is Mismatch breakpoint, not a match breakpoint
N unsigned int
Breakpoint number
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_t32_vector_catch

Fields:

ADDR unsigned int
Address
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_watchpoint_far

Fields:

LOWER_BOUND unsigned int
the lowest address accessed by the instruction that triggered the watchpoint
UPPER_BOUND unsigned int
the highest watchpointed address accessed by that instruction

ArchMsg.Warning.warning_AdvSIMDExpandImmUnexpectedZero

AdvSIMDExpandImm may treat this immediate value as UNPREDICTABLE.

ArchMsg.Warning.warning_ConditionalSMC

SMC instruction has UNPREDICTABLE effects when conditional when combined with traps.

ArchMsg.Warning.warning_access_crosses_page_boundary

Access crosses page boundary.

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault_on_both_pages

Access crosses page boundary has fault(s).

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_access_crosses_page_boundary_spanning_different_memory

Fields:

ADDR unsigned int
: address of access crossing page

MEMTYPE_PAGE1 signed int
: First Page's memory type (0-Normal, 1-Non-Normal)

MEMTYPE_PAGE2 signed int
: Second Page's memory type (0-Normal, 1-Non-Normal)

ArchMsg.Warning.warning_access_wraps_around_memory

Fields:

WIDTH unsigned int
: Address width

ArchMsg.Warning.warning_bx_from_thumbee

Fields:

ADDR unsigned int
: Destination address

ArchMsg.Warning.warning_ccsidr_unimplemented_level

Fields:

CSSELR unsigned int
: current effective value of CSSELR

ArchMsg.Warning.warning_change_to_ns_when_tge_set

Attempting to change NS to 1 when HCR.TGE = 1.

ArchMsg.Warning.warning_contiguous_bit_check_abort

Fields:

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ArchMsg.Warning.warning_contiguous_bit_error

Fields:

CONTIG_ADDR unsigned int
: address of the TLB first read, with which this entry is expected to be contiguous

CONTIG_DATA unsigned int
: expected contents based on the entry at CONTIG_ADDR

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ENTRY_DATA unsigned int
: contents of conflicting TLB entry

ArchMsg.Warning.warning_cp10_cp11_mismatch

Fields:

EL unsigned int
: EL owning the control

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_cp10_cp11_reserved_value

Fields:

CP10 unsigned int
: CP10 access

CP11 unsigned int
: CP11 access

ArchMsg.Warning.warning_csselr_level_out_of_range

Fields:

IMPLEMENTED unsigned int
: number of cache levels implemented

LEVEL unsigned int
: written value of CSSELR.Level

TYPE enum
: The type of cache selected by CSSELR.InD and TnD

ArchMsg.Warning.warning_debug_flow_control_bits_not_obeyed

Fields:

ERROR enum
: type of error resulting

REGISTER enum
: register under consideration

ArchMsg.Warning.warning_debug_register_access_during_reset

Fields:

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

ArchMsg.Warning.warning_decode_cps_inconsistent_fields

The mode setting fields in the CPS instruction are inconsistent.

ArchMsg.Warning.warning_decode_invalid_state

Instruction is unpredictable in current exception-level/security state.

ArchMsg.Warning.warning_default_cacheable_mmu_on

Default cacheable is enabled (HCR.DC = 1) while MMU is enabled.

ArchMsg.Warning.warning_default_cacheable_vmmu_off

Default cacheable is enabled (HCR.DC = 1) while HCR.VM = 0.

ArchMsg.Warning.warning_deprecated_wvr_bit_2_set

Fields:

N unsigned int
: index

WVR unsigned int
: DBGWVR

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_eret_while_software_step_active_pending

Missing ISB between setting MDSCR_EL1.SS and ERET.

ArchMsg.Warning.warning_exclusive_to_non_normal

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_exclusive_to_non_writeback

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_execute_from_device_memory

An attempt was made to execute from Device memory.

ArchMsg.Warning.warning_illegal_cpsr_mode

Fields:

NEW_MODE unsigned int
: New value of CPSR.M

ArchMsg.Warning.warning_illegal_srs_mode

Fields:

MODE unsigned int
: mode for Banked SP

ArchMsg.Warning.warning_implementation_defined_read_debug_register_in_SCS

Read Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_implementation_defined_sequential_security_transitions_supported

The behaviour of a sequential instruction fetches that cross from non-secure to secure memory and contain SG instruction is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_implementation_defined_stack_limit_check_supported

It is IMPLEMENTATION DEFINED whether stack pointer limit checking is performed for the this instructions. Fields:

ADDRESS unsigned int
: Address of instruction

OPCODE unsigned int
: instruction opcode

ArchMsg.Warning.warning_implementation_defined_write_debug_register_in_SCS

Write Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

DATA unsigned int
: data attempted to be written

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PPB_OFFSET unsigned int
debug register offset

ArchMsg.Warning.warning_invalid_tcr_granule

Fields:

REQUEST signed int
page size requested (or 0 for reserved)
SUBSTITUTE signed int
best guess available page size
TG_ID bool
bits TG0 or TG1

ArchMsg.Warning.warning_load_multiple_user_registers_from_user_mode

An LDM(user registers) instruction executed from user mode.

ArchMsg.Warning.warning_load_pc_from_unaligned

PC loaded from an unaligned location.

ArchMsg.Warning.warning_mte_in_mem_attr_not_supported

Fields:

ATTR unsigned int
Memory attribute

ArchMsg.Warning.warning_reserved_breakpoint_state_match

Fields:

IS_BREAKPOINT unsigned int
Is breakpoint, not watchpoint

ArchMsg.Warning.warning_shareability

Unpredictable: combination of the 1st and 2nd stages of translation is Normal Inner Non-Cacheable, Outer Non-Cacheable.

ArchMsg.Warning.warning_software_step_set_while_enabled

MDSCR_EL1.SS set to 1 while software step debug exceptions are enabled.

ArchMsg.Warning.warning_thumb_instruction_wraps_around_memory

Thumb instruction warps around memory.

ArchMsg.Warning.warning_ttbr_sbz_bits_are_not_zero

Fields:

MSB unsigned int
x-1
TTBR enum
which TTBR register

ArchMsg.Warning.warning_unaligned_address_dbgdtrrx_write

Fields:

ADDR unsigned int
Address

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unaligned_address_dbgdtrtx_read

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_device

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_strongly_ordered

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unknown_sau_rnr

SAU_RNR was set to an unsupported value.

ArchMsg.Warning.warning_unpredictable_AIRCR_PRIP_and_BFHFNMINP

The effect of setting both AIRCR.BFHFNMINP and AIRCR.PRIP to 1 is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTCLRACTIVE_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTCLRACTIVE if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_and_SYSRESETREQ

When the processor is halted in Debug state, if a write to the register writes a 1 to both VECTRESET and SYSRESETREQ, the behavior is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTRESET if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_incorrect_VKEY

The value 0x05FA must be written to AIRCR.VKEY, otherwise the register write is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_EXE_RETURN_Reserved_Bit

EXC_RETURN[23:7] are reserved with the special condition that all bits should be written as one. Values other than all 1s are UNPREDICTABLE. Fields:

EXC_RETURN unsigned int
Special PC value
SBO_MASK unsigned int
Which bits Should Be One to not UNPRED

ArchMsg.Warning.warning_unpredictable_change_to_DEMCR_MON_STEP_at_insufficient_priority

The effect of changing DEMCR.MON_STEP at an execution priority that is lower than the priority of the DebugMonitor exception is CONSTRAINED UNPREDICTABLE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_change_to_priority_of_active_exception

Changing the priority of an active exception is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_clear_lspact

Fields:

DRVEXC_TAKEN unsigned int
: derived exception taken not pended

REACHED_LAST_STORE unsigned int
: only the last faulted

ArchMsg.Warning.warning_unpredictable_dbg_exit_unaligned_dlr

Debug exit to AArch32 with DLR[0] set to 1. PC[0] can be set to 0 or to DLR[0].

ArchMsg.Warning.warning_unpredictable_exception_catch

Generation of an exception catch event is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_inconsistent_state

state on exception return is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_instruction

exception return instruction is unpredictable.

ArchMsg.Warning.warning_unpredictable_in_debug_state

Instruction is UNPREDICTABLE when executed in debug state.

ArchMsg.Warning.warning_unpredictable_mair_encoding

Fields:

ATTR unsigned int
: Memory attribute

ArchMsg.Warning.warning_unpredictable_pmu_counter_access

Access to PMU counters from non-secure EL0 or EL1 is UNPREDICTABLE with MDCR_EL2.HPMN set to 0.

ArchMsg.Warning.warning_unpredictable_prioritization_breakpoint_match_vector_catch

Two events with the same priority occurred at the same instruction: Address Matching Vector Catch debug event, and Breakpoint debug event It is constrained unpredictable which is taken.

ArchMsg.Warning.warning_unpredictable_stack_selection

UNPREDICTABLE state on context switch.

ArchMsg.Warning.warning_unpredictable_tsize_out_of_range

Fields:

TCR enum
: Which TCR

TSIZE signed int
: Value of TSize

TSIZE_MAX signed int
: The maximum allowed value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

TTBR unsigned int
Which TTBR region

ArchMsg.Warning.warning_unpredictable_unaligned_pc_as_base_register

The PC value must be word-aligned, otherwise the behavior of the instruction is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_unaligned_pop_stack

Fields:

FRAME_ADDR unsigned int
unaligned stack frame address

ArchMsg.Warning.warning_unpredictable_vmsa_memattrib

Fields:

TEXCB unsigned int
texcb value
TEX_REMAP bool
use tex remap

ArchMsg.Warning.warning_unpredictable_vtcr_t0sz_sl0

Fields:

SL0 signed int
VTCR.SL0, starting level for stage 2 translation table walks
T0SZ signed int
VTCR.T0SZ, required input address range

ArchMsg.Warning.warning_unpredictable_write_DHCSR

Fields:

NEW unsigned int
data attempted to be written
OLD unsigned int
previous value

ArchMsg.Warning.warning_unpredictable_write_SHCSR

UNPREDICTABLE setting a pending bit to 1 for an exception with priority >= execution priority.

ArchMsg.Warning.warning_unsupported_access_to_memory_mapped_register

Access to memory mapped register is unsupported.

ArchMsg.Warning.warning_user_jmcr_access

Fields:

WRITE unsigned int
access is write

ArchMsg.Warning.warning_wcr_mask_and_bas

DBGWCR_EL1.MASK is non-zero and BAS is not 0xff.

ArchMsg.Warning.warning_wcr_mask_reserved

DBGWCR_ELn.MASK is set to a reserved value.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_wcr_non_configuous_bas

DBGWCR_EL1.BAS has non-contiguous set of ones.

ArchMsg.Warning.warning_wvr_masked_bit_not_zero

DBGWVRn_EL1.VA contains a bit masked by DBGWCRn_EL1.MASK that is not zero.

ArchMsg.Why.why_branch_target_exception

A branch target exception has occurred. Fields:

BTYPE unsigned int
: pstate.BTYPE at time branch taken

OPCODE unsigned int
: opcode at branch target

PC unsigned int
: Virtual Address of branch target

ArchMsg.Why.why_illegal_state

An illegal mode change has occured. Fields:

MESSAGE string
: The message

BRANCH_MISPREDICT

Simulating branch mispredict. Fields:

PC unsigned int
: Origin address (or 0 if unavailable)

BRA_DIR

Direct branches, to immediate address. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR

Indirect branches, perhaps to a register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR_CCFAIL

Indirect branches fail, perhaps to a register.

BRA_RET_CCFAIL

Branche return fail, perhaps to a register.

CACHE_MAINTENANCE_OP

Cache Maintenance Operation. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Specified MVA or Set/way

FUNCTION enum
: Clean / Invalidate

SCOPE enum
: Affected region

SIDE enum
: Inst / Data.

CCFAIL

Conditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCFAIL_UNC

Unconditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS

Conditional instruction condition check pass. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS_UNC

Unconditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CHECKPOINT_MESSAGE

Report error messages from the checkpointing process. Fields:

message string
: Message contents.

CHECKPOINT_RESTORE_END

Checkpoint restore completed.

CHECKPOINT_RESTORE_START

Checkpoint restore about to start.

CHECKPOINT_SAVE_END

Checkpoint save completed.

CHECKPOINT_SAVE_START

Checkpoint save about to start.

CODE_CACHE_MAINT

Code cache maintenance. Fields:

END_INCL unsigned int
: Inclusive end addres for by VA or PA requests

NS bool
: Non-secure world for PA

START unsigned int
: Start address for by VA or PA requests

TYPE enum
: Request type.

COMPILE_BLOCK_END

Last instruction of basic block translated. Fields:

VADDR unsigned int
: Address of next instruction after this basic block

COMPILE_BLOCK_START

First instruction of basic block translated. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VADDR unsigned int
: Address of first instruction in basic block

COMPILE_INST

ARM instruction compiled. Fields:

DISASS string
: Disassembly of the instruction.

ISET enum
: The instruction set of this instruction.

ITSTATE unsigned int
: The ITSTATE current for the instruction.

OPCODE unsigned int
: The opcode of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

CONTEXTIDR

Context ID Register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

NS bool
: Secure or nonsecure banked register is accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

CONTEXT_SYNC

Called for every context synchronization event. Fields:

ARCH_CSE bool
: Architectural Context Synchronization Event

SYSREG_SYNC bool
: System register synchronization event

CORE_ENDIAN

Core BE8 Big-Endian state changed. Fields:

BE8 bool
: Core BE8 Big-Endian state

CORE_INFO

Static processor attributes. Only triggered by a call to DumpState(). Fields:

ARCH_PROFILE enum
: The architecture profile of the core.

CLUSTER_ID unsigned int
: The cluster ID of this processor.

CORE_NUM unsigned int
: The number of this core in an MP processor.

DETAILED_CORE_FEATURES enum
: Detailed core features as a bitfield. Enum values are bitmasks to extract the info from the field

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

FPU_VERSION enum
: The VFP version implemented by the core.

MEM_ARCH enum
: The memory architecture of the core.

MIDR_EL1 unsigned int
: MIDR_EL1 value

NUM_CORES unsigned int
: The number of cores in this MP processor.

QUANTUM_SIZE unsigned int
: The default quantum size of the core.

SECURITY_FEATURES bool
: Does the core have security features?

CORE_LOADS

Processor load accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_REGS

Changes of the core registers R0 to R14. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number, 0 to 14.

MODE enum
: Bank of the register accessed.

OLD_VALUE unsigned int
: The old value overwritten.

PHYS_ID enum
: The physical register accessed.

VALUE unsigned int
: The new value written to the register.

CORE_REGS64

Changes of the core registers X0..X30, SP_ELn. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: The register number, 0..30 for X0..X30, >=32 for SP_ELn.

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

CORE_STORES

Processor store accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CP14_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP14_WRITE

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

CP15_WRITE

System Control Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

CPSR

CPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

OLD_VALUE unsigned int
: The old CPSR value

UNKNOWN unsigned int
: Bits within the register that have unknown value.

VALUE unsigned int
: The new CPSR value

CRYPTO_SPEC

Every crypto instruction speculatively executed.

CorePowerStateChange

Core power-up state has changed. Fields:

PU bool
: Whether the core powered up

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DATA_CACHE_SET_ALLOC_TAGS

Set allocation tags from DC GVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of memory in bytes.

TAG unsigned int
: The value of the tag being set.

VADDR unsigned int
: Virtual address used for setting allocation tags.

DATA_CACHE_ZERO

Zero and invalidate cache line from DC ZVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of zero'd memory in bytes.

VADDR unsigned int
: Virtual address of the zero'd cache line.

DEBUG_EVENT

Hardware debug support event. Fields:

EVENT enum
: Description of event

VALUE unsigned int
: data value

END_COMPILE

Compilation end. Fields:

END_OF_PAGE_COUNT unsigned int
: Number of basic blocks exited due to page boundary since START_COMPILE.

FETCHFAIL_COUNT unsigned int
: Number of basic blocks exited due to ifetch failure START_COMPILE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: Number of instructions compiled since START_COMPILE.

NONSEQ_COUNT unsigned int
: Number of basic blocks exited due to non-sequential instructions since START_COMPILE.

PAGE_STRADDLE_COUNT unsigned int
: Number of basic blocks exited due to unaligned instructions crossing page since START_COMPILE.

ESB

Error Synchronization Barrier hit. Fields:

ASYNC bool
: ESB is async

IMPLICIT bool
: ESB is implicit

EXCEPTION

Exception taken. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

IS_PHYSICAL bool
: Physical or Virtual exception

LR unsigned int
: The value assigned to the link register.

PC unsigned int
: The location where the exception occurred.

PREFERRED_RETURN unsigned int
: The preferred return address for the exception.

TARGET_ISET enum
: The instruction set of the exception handler code.

TARGET_PC unsigned int
: The address the exception branches to.

VECTOR enum
: The exception vector.

VECTOR_OFFSET unsigned int
: The offset of an exception vector from the base address.

EXCEPTION_END

Every exception completed.

EXCEPTION_RAISE

Every exception raised.

EXCEPTION_RETURN

Branches on leaving exception. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

EXCEPTION_RETURN_PREBRANCH

Exception return event that must occur before PC and processor state have updated.

EXCEPTION_START

Every exception started.

FIQ_TAKEN

FIQ taken exception.

FP_STATE

Floating point state. Fields:

FPSR unsigned int
: FPSR register value.

MASK unsigned int
: The mask indicating updated cumulative bits.

HLT

HLT instruction occurred. Fields:

IMM16 unsigned int
: The 16-bit immediate value encoded in HLT instruction.

INST

Every instruction executed. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DEBUG_STATE bool
: The instruction is executed in debug state.

DISASS string
: Disassembly of instruction.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The current instruction set.

ITSTATE unsigned int
: The current ITSTATE.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

MODE enum
: The mode the core is in.

NS unsigned int
: The core's non-secure bit.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

OPCODE unsigned int
: The opcode of the instruction.

PADDR unsigned int
: The physical address of the instruction.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

INST_END

Every instruction completed.

INST_START

Every instruction started. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

INST_STRADDLE

Instruction straddles boundary. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

IRQ_TAKEN

IRQ taken exception.

ISB

ISB instruction occurred.

LOAD_MULTIPLE_REGS_LIST

List of destination registers (not source) of a multiple registers load instruction. Fields:

REG string
: Register to add into the list

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOCAL_MONITOR

Local monitor activity. Fields:

PADDR unsigned int
: Local Monitor Address

State enum
: State of the monitor (Open/Exclusive)

MEMMAP_DEBUG_READ

Memory mapped reads to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

MEMMAP_DEBUG_WRITE

Memory mapped writes to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

MMU_TRANS

Address translation information. Fields:

ASID unsigned int
: Address space identifier.

CORE_NUM unsigned int
: Core number in a multi processor.

Hyp bool
: Entry matches in Hyp state only

INNERCACHE_RA bool
: Is the inner cache allocate on read

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INNERCACHE_TRANSIENT bool
: Is the inner write-through transient

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

MEMTYPE enum
: Memory type.

NSDESC enum
: Is secure side supposed to access secure or nonsecure memory

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TRANSIENT bool
: Is the outer write-through transient

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

PADDR unsigned int
: Physical address of the access.

PAGESIZE unsigned int
: Page size as log2(size).

SH enum
: Shareability

SIDE enum
: Inst / Data.

STAGE1_PERM unsigned int
: Stage 1 permissions, [5:3] Privileged access (XWR) [2:0] User access (XWR)

STAGE2_PERM unsigned int
: Stage 2 permission mask [5:3] Privileged access (XWR) [2:0] User access (XWR)

VADDR unsigned int
: Virtual address of the access.

VMID unsigned int
: Virtual machine identifier.

XS bool
: XS Attribute

nG enum
: Flag indicating whether ASID will be matched

MMU_TTB_READ

This event is triggered by reads caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data read.

IPA unsigned int
: For stage 1, the IPA of the read.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OUTADDR unsigned int
: The output physical address.

PADDR unsigned int
: The physical address of the read.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MMU_TTB_WRITE

This event is triggered by writes caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data written.

IPA unsigned int
: For stage 1, the IPA of the write.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

PADDR unsigned int
: The physical address of the write.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MODE_CHANGE

Mode change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: The new mode.

NON_SECURE enum
: The new security state

OLD_MODE enum
: The old mode.

NOT_TAG_CHECKED_INST

MTE Tag not checked for current instruction.

PERIODIC

Called for every quantum. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PC unsigned int
The address of the next instruction to be executed on this CPU.

PMU_COUNTER_OVERFLOW

PMU counter overflow. Fields:

EVENT_ID unsigned int
Event Identifier
INDEX unsigned int
Counter index (as selected by PMSELR).
INTERRUPT bool
Is interrupt enabled on overflow for this counter

PREFETCH_MEMORY64

Prefetch from PRFM or PRFUM instructions. Fields:

PRFOP unsigned int
Prefetch hint
VADDR unsigned int
Virtual address of the location that should be prefetched.

PRELOAD_DATA

Data preload from PLD instruction. Fields:

VADDR unsigned int
Virtual address of the data that should be preloaded.

PRELOAD_INST

Instruction preload from PLI instruction. Fields:

VADDR unsigned int
Virtual address of the instruction that should be preloaded.

PRE_CORE_LOAD

Trace just before a core load. Fields:

CURRENT_TIME unsigned int
The core's current time, as simulated time plus local time.
LOCAL_TIME unsigned int
The core's local time, relative to the current quantum.

PRE_CORE_STORE

Trace just before a core store. Fields:

CURRENT_TIME unsigned int
The core's current time, as simulated time plus local time.
LOCAL_TIME unsigned int
The core's local time, relative to the current quantum.

PRE_TTB_READ

This event is triggered before reads caused by a translation table walk. Fields:

ADDR unsigned int
The physical address of the read.

PSTATE_PACTIVE_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions on PChannel. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

PSTATE_PACTIVE_OPMODE_TRANSITION_DENIED

Trace the denied pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_POWER_TRANSITION

Trace pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

PSTATE_PACTIVE_POWER_TRANSITION_DENIED

Trace the denied pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_THREAD_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions for threads on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

RUN_STATE

Run state transition. Fields:

INST_COUNT unsigned int
: Ticks count at point of transition.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NEW enum
: New run state.

OLD enum
: Old run state.

SEMIHOSTING_CALL

Call of a semihost function occurred. Fields:

PC unsigned int
: The program counter after the semihosting call.

SEMIHOSTING_PRECALL

About to call semihost. Fields:

EL2 bool
: Use EL2 translation regime.

NS bool
: Is Non-Secure.

REG_WIDTH unsigned int
: The current register width in bytes.

SIGNAL

External signal state change. Fields:

SIGNAL enum
: Signal that changed

STATE bool
: Signal asserted state

SOFTWARE_STEP

Return the Debug Software Step state. Fields:

STATE enum
: Software Step state

SPSR

SPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: Which of the banked SPSR registers is written.

OLD_VALUE unsigned int
: The old SPSR value

VALUE unsigned int
: The new SPSR value

START_COMPILE

Compilation started. Fields:

VADDR unsigned int
: Instruction where compilation begins.

STG_MULTIPLE_MICRO_OPS_ADDRESS_RANGE

Address range for MTE stores with multiple uops. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

STORE_MULTIPLE_REGS_ADDRESS_RANGE

Address range for stores with multiple registers. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

SUBOPTIMAL_LDST_RETIRED

suboptimal load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1: half-word, 2: word ...)

OPTIMAL_ALIGN_SELECTOR unsigned int
: Selector to decide the optimal alignment size (1: access width, 2: 4-bytes, 3: 8-bytes, ..., 12: 4-kbytes)

VADDR unsigned int
: The virtual address of the access

SYNC

Called for every synchronization, that is every quantum break. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

LOCAL_QUANTUM unsigned int
: The local quantum of this CPU.

LOCAL_TIME unsigned int
: The local time of this CPU.

SYNC_REASON enum
: Reason for sync.

SYSCALL

System call instruction executed. Fields:

IMM unsigned int
: Immediate value of the system call instruction.

TYPE enum
: System call type.

VADDR unsigned int
: Instruction that caused the system call.

SYSREG_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE32

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE64

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

UNALIGNED_LDST_RETIRED

Processor unaligned load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1=>half-word, 2->word ...)

NUMBER_OF_BEATS unsigned int
: Number of accesses (beats) in this burst (Total size in bytes is ACCESS_SIZE * NUMBER_OF_BEATS)

VADDR unsigned int
: The virtual address of the access

VFP_D_REGS

VFP/NEON D 64 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_Q_REGS

VFP/NEON Q 128 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding byte in the register. For example, 0x1 would indicate a write to VALUE[7:0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_SYS_REGS

Writes to the VFP/NEON units system registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: Which VFP system register is written.

MASK unsigned int
: The mask indicating updated cumulative bits.

OLD_VALUE unsigned int
: The register's old value overwritten.

VALUE unsigned int
: The new value written to the VFP system register.

VFP_S_REGS

VFP/NEON S 32 bit register write. Fields:

ALIASING enum
: Type of register aliasing used.

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

WAYPOINT

Signals end of basic block execution. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: Origin instructions set

IS_COND bool
: Indicates if this is a conditional waypoint.

PC unsigned int
: Origin address (or 0 if unavailable)

TAKEN bool
: Indicates if this waypoint was taken.

TARGET unsigned int
: Destination address

TARGET_ISET enum
: Destination instructions set

WFE_END

WFE ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFE.

WFE_EVENT_REGISTER

WFE event register status: set/clear, reason. Fields:

INST_COUNT unsigned int
: Ticks count.

REASON enum
: Reason for set/clear. Only REASON==0 (Cleared by WFE) clears the bit, all other reasons set it.

WFE_IGNORED

WFE ignored. Fields:

EVENT bool
: This WFE was ignored because the event register was set.

INST_COUNT unsigned int
: Ticks count when ignoring WFE.

TRAPPED bool
: This WFE was trapped.

WFE_START

WFE entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFE.

WFI_END

WFI ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFI.

WFI_IGNORED

WFI ignored. Fields:

DISABLED bool
: This WFI was ignored because WFI is disabled.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: Ticks count when ignoring WFI.

REASON enum
: specifies reason why WFI trace was ignored

TRAPPED bool
: This WFI was trapped.

WFI_START

WFI entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFI.

WFI_WAKEUP

WFI wakeup. Fields:

INST_COUNT unsigned int
: Ticks count when WFI wakeup occurred.

REASON enum
: Reason for wakeup.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.19 ARM_Cortex-A73 - trace

This section describes the trace sources.

ASYNC_MEMORY_FAULT

Context ID Register write. Fields:

FAULT unsigned int
: Fault status in ESR format

PADDR unsigned int
: Physical Address (or 0 if unavailable)

VADDR signed int
: Virtual Address (or 0 if unavailable)

ATOMIC_END_ACCESS

Bus trace access for atomics. Fields:

ACCESS_FAIL bool
: Memory access failed

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

LOAD_VALUE unsigned int
: Loaded values

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ATOMIC_START_ACCESS

Bus trace access for atomics. Fields:

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

NSDESC unsigned int
: The security state of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ArchMsg.Error.exit_code

Fields:

EXITCODE signed int
: Exit code to be returned

KIND string
: Component kind that invoked the exit code trace

REASON string
: Reason why the exit code trace was invoked (optional)

ArchMsg.Warning.IgnoredWarmResetRequest

Event fires when software attempts to request warm reset through write to deprecated DBGPRCR.CWRR.

ArchMsg.Warning.TarmacReconstructor

Unexpected event happened during Tarmac reconstruction. Fields:

REASON string
: Reason

ArchMsg.Warning.Unpredictably Indexed PM Event Register

Fields:

IsDirect unsigned int
: Direct Access rather than indirect

N unsigned int
: Number of Accessible Registers

SEL unsigned int
: Selector

ArchMsg.Warning.branch_to_unaligned_address

Fields:

ADDR unsigned int
: Unaligned branch address

ArchMsg.Warning.cache_contents_unknown

A cache has been activated without being invalidated and may contain UNKNOWN content. Fields:

REGIME unsigned int
: Current translation regime

SIDE unsigned int
: 0 d-side cache or unified cache, 1 i-side

ArchMsg.Warning.dcimvac_matches_watchpoint

Watchpoints matching an AArch32 DCIMVA are implementation defined.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_fieldmismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_fracbitsnegative

VCVT Instruction is unpredictable with a negative number of fraction bits.

ArchMsg.Warning.decode_initblock

Instruction is unpredictable in an IT block.

ArchMsg.Warning.decode_initblocknotlast

Instruction is unpredictable in an IT block when not the last.

ArchMsg.Warning.decode_invalid_condition

Instruction is unpredictable when the condition is not AL.

ArchMsg.Warning.decode_invalidfieldcombination

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_invalidvalue

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_mem_hint_unallocated

There is no memory hint allocated to this bit pattern.

ArchMsg.Warning.decode_registermatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registermismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registeroutofrange

Float point instruction is unpredictable indexing beyond the end of the register bank.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_sbzsbo

Reserved bits in the instruction are not canonical.

ArchMsg.Warning.decode_transactiontoobig

Attempt to perform a memory transaction of more than 128 bytes.

ArchMsg.Warning.decode_unpred_other

Instruction is unpredictable.

ArchMsg.Warning.decode_unpreduseofpc

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13inreglist

Instruction is unpredictable with R13 in register-list.

ArchMsg.Warning.decode_useofd16orhigher

Attempt to use register D16 or higher where it is not allowed.

ArchMsg.Warning.decode_writebackandbaseinlist

Instruction is unpredictable with write back when the base register is in the transfer list.

ArchMsg.Warning.decode_zeroregistersinlist

Instruction is unpredictable when no registers are to be transfered.

ArchMsg.Warning.recursive_branch

An instruction is performing a branch that targets the same instruction. If this was intended then a WFI might be more effective. Fields:

PC unsigned int
: Address of instruction causing the branch

ArchMsg.Warning.recursive_exception

An instruction has generated an exception that targets the same instruction. Fields:

CPSR unsigned int
: Processor state

PC unsigned int
: Address of instruction causing the exception

TYPE enum
: Type of exception

ArchMsg.Warning.reentrant_vector_catch

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR unsigned int
vector address

ArchMsg.Warning.reserved_it_state

Fields:

ITSTATE unsigned int
reserved IT state

ArchMsg.Warning.tlb_contents_unknown

Translation has been enabled without invalidating the TLB. Fields:

EL unsigned int
Exception level of translation regime
INVALIDITY unsigned int
Which TLB may be invalid (if TLBs are separate)
NON_SECURE unsigned int
Security state of translation regime
VMID unsigned int
Current VMID, if applicable

ArchMsg.Warning.unknown_DLR_DSPSR

Exiting debug state with unknown DLR or DSPSR.

ArchMsg.Warning.unknown_ELR_SPSR

Returning from debug exception with unknown ELR or SPSR.

ArchMsg.Warning.unpredictable_a32_breakpoint

Fields:

ADDR1 unsigned int
Address
ADDR2 unsigned int
Breakpoint address
BAS unsigned int
BAS Field
IS_ADDRESS_MISMATCH_BREAKPOINT bool
Is Mismatch breakpoint, not a match breakpoint
N unsigned int
Breakpoint number
OFFSET unsigned int
Indicates which halfword actually considering into the A32 instruction

ArchMsg.Warning.unpredictable_banked_register_access

Fields:

R unsigned int
R bit
SYSM unsigned int
SYSm

ArchMsg.Warning.unpredictable_ccfail_undef

Fields:

OPCODE unsigned int
opcode of the instruction

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VADDR unsigned int
Virtual address of the instruction being executed

ArchMsg.Warning.unpredictable_ici_bits

ICI bits do not apply or are architecturally unpredictable for instruction.

ArchMsg.Warning.unpredictable_t32_breakpoint

Fields:

ADDR1 unsigned int
Address
ADDR2 unsigned int
Breakpoint address
BAS unsigned int
BAS Field
IS_ADDRESS_MISMATCH_BREAKPOINT bool
Is Mismatch breakpoint, not a match breakpoint
N unsigned int
Breakpoint number
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_t32_vector_catch

Fields:

ADDR unsigned int
Address
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_watchpoint_far

Fields:

LOWER_BOUND unsigned int
the lowest address accessed by the instruction that triggered the watchpoint
UPPER_BOUND unsigned int
the highest watchpointed address accessed by that instruction

ArchMsg.Warning.warning_AdvSIMDExpandImmUnexpectedZero

AdvSIMDExpandImm may treat this immediate value as UNPREDICTABLE.

ArchMsg.Warning.warning_ConditionalSMC

SMC instruction has UNPREDICTABLE effects when conditional when combined with traps.

ArchMsg.Warning.warning_access_crosses_page_boundary

Access crosses page boundary.

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault_on_both_pages

Access crosses page boundary has fault(s).

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_access_crosses_page_boundary_spanning_different_memory

Fields:

ADDR unsigned int
: address of access crossing page

MEMTYPE_PAGE1 signed int
: First Page's memory type (0-Normal, 1-Non-Normal)

MEMTYPE_PAGE2 signed int
: Second Page's memory type (0-Normal, 1-Non-Normal)

ArchMsg.Warning.warning_access_wraps_around_memory

Fields:

WIDTH unsigned int
: Address width

ArchMsg.Warning.warning_bx_from_thumbee

Fields:

ADDR unsigned int
: Destination address

ArchMsg.Warning.warning_ccsidr_unimplemented_level

Fields:

CSSELR unsigned int
: current effective value of CSSELR

ArchMsg.Warning.warning_change_to_ns_when_tge_set

Attempting to change NS to 1 when HCR.TGE = 1.

ArchMsg.Warning.warning_contiguous_bit_check_abort

Fields:

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ArchMsg.Warning.warning_contiguous_bit_error

Fields:

CONTIG_ADDR unsigned int
: address of the TLB first read, with which this entry is expected to be contiguous

CONTIG_DATA unsigned int
: expected contents based on the entry at CONTIG_ADDR

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ENTRY_DATA unsigned int
: contents of conflicting TLB entry

ArchMsg.Warning.warning_cp10_cp11_mismatch

Fields:

EL unsigned int
: EL owning the control

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_cp10_cp11_reserved_value

Fields:

CP10 unsigned int
: CP10 access

CP11 unsigned int
: CP11 access

ArchMsg.Warning.warning_csselr_level_out_of_range

Fields:

IMPLEMENTED unsigned int
: number of cache levels implemented

LEVEL unsigned int
: written value of CSSELR.Level

TYPE enum
: The type of cache selected by CSSELR.InD and TnD

ArchMsg.Warning.warning_debug_flow_control_bits_not_obeyed

Fields:

ERROR enum
: type of error resulting

REGISTER enum
: register under consideration

ArchMsg.Warning.warning_debug_register_access_during_reset

Fields:

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

ArchMsg.Warning.warning_decode_cps_inconsistent_fields

The mode setting fields in the CPS instruction are inconsistent.

ArchMsg.Warning.warning_decode_invalid_state

Instruction is unpredictable in current exception-level/security state.

ArchMsg.Warning.warning_default_cacheable_mmu_on

Default cacheable is enabled (HCR.DC = 1) while MMU is enabled.

ArchMsg.Warning.warning_default_cacheable_vmmu_off

Default cacheable is enabled (HCR.DC = 1) while HCR.VM = 0.

ArchMsg.Warning.warning_deprecated_wvr_bit_2_set

Fields:

N unsigned int
: index

WVR unsigned int
: DBGWVR

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_eret_while_software_step_active_pending

Missing ISB between setting MDSCR_EL1.SS and ERET.

ArchMsg.Warning.warning_exclusive_to_non_normal

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_exclusive_to_non_writeback

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_execute_from_device_memory

An attempt was made to execute from Device memory.

ArchMsg.Warning.warning_illegal_cpsr_mode

Fields:

NEW_MODE unsigned int
: New value of CPSR.M

ArchMsg.Warning.warning_illegal_srs_mode

Fields:

MODE unsigned int
: mode for Banked SP

ArchMsg.Warning.warning_implementation_defined_read_debug_register_in_SCS

Read Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_implementation_defined_sequential_security_transitions_supported

The behaviour of a sequential instruction fetches that cross from non-secure to secure memory and contain SG instruction is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_implementation_defined_stack_limit_check_supported

It is IMPLEMENTATION DEFINED whether stack pointer limit checking is performed for the this instructions. Fields:

ADDRESS unsigned int
: Address of instruction

OPCODE unsigned int
: instruction opcode

ArchMsg.Warning.warning_implementation_defined_write_debug_register_in_SCS

Write Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

DATA unsigned int
: data attempted to be written

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PPB_OFFSET unsigned int
debug register offset

ArchMsg.Warning.warning_invalid_tcr_granule

Fields:

REQUEST signed int
page size requested (or 0 for reserved)
SUBSTITUTE signed int
best guess available page size
TG_ID bool
bits TG0 or TG1

ArchMsg.Warning.warning_load_multiple_user_registers_from_user_mode

An LDM(user registers) instruction executed from user mode.

ArchMsg.Warning.warning_load_pc_from_unaligned

PC loaded from an unaligned location.

ArchMsg.Warning.warning_mte_in_mem_attr_not_supported

Fields:

ATTR unsigned int
Memory attribute

ArchMsg.Warning.warning_reserved_breakpoint_state_match

Fields:

IS_BREAKPOINT unsigned int
Is breakpoint, not watchpoint

ArchMsg.Warning.warning_shareability

Unpredictable: combination of the 1st and 2nd stages of translation is Normal Inner Non-Cacheable, Outer Non-Cacheable.

ArchMsg.Warning.warning_software_step_set_while_enabled

MDSCR_EL1.SS set to 1 while software step debug exceptions are enabled.

ArchMsg.Warning.warning_thumb_instruction_wraps_around_memory

Thumb instruction warps around memory.

ArchMsg.Warning.warning_ttbr_sbz_bits_are_not_zero

Fields:

MSB unsigned int
x-1
TTBR enum
which TTBR register

ArchMsg.Warning.warning_unaligned_address_dbgdtrrx_write

Fields:

ADDR unsigned int
Address

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unaligned_address_dbgdtrtx_read

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_unaligned_to_device

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_unaligned_to_strongly_ordered

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_unknown_sau_rnr

SAU_RNR was set to an unsupported value.

ArchMsg.Warning.warning_unpredictable_AIRCR_PRIP_and_BFHFNMINP

The effect of setting both AIRCR.BFHFNMINP and AIRCR.PRIP to 1 is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTCLRACTIVE_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTCLRACTIVE if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_and_SYSRESETREQ

When the processor is halted in Debug state, if a write to the register writes a 1 to both VECTRESET and SYSRESETREQ, the behavior is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTRESET if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_incorrect_VKEY

The value 0x05FA must be written to AIRCR.VKEY, otherwise the register write is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_EXE_RETURN_Reserved_Bit

EXC_RETURN[23:7] are reserved with the special condition that all bits should be written as one. Values other than all 1s are UNPREDICTABLE. Fields:

EXC_RETURN unsigned int
: Special PC value

SBO_MASK unsigned int
: Which bits Should Be One to not UNPRED

ArchMsg.Warning.warning_unpredictable_change_to_DEMCR_MON_STEP_at_insufficient_priority

The effect of changing DEMCR.MON_STEP at an execution priority that is lower than the priority of the DebugMonitor exception is CONSTRAINED UNPREDICTABLE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_change_to_priority_of_active_exception

Changing the priority of an active exception is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_clear_lspact

Fields:

DRVEXC_TAKEN unsigned int
: derived exception taken not pended

REACHED_LAST_STORE unsigned int
: only the last faulted

ArchMsg.Warning.warning_unpredictable_dbg_exit_unaligned_dlr

Debug exit to AArch32 with DLR[0] set to 1. PC[0] can be set to 0 or to DLR[0].

ArchMsg.Warning.warning_unpredictable_exception_catch

Generation of an exception catch event is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_inconsistent_state

state on exception return is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_instruction

exception return instruction is unpredictable.

ArchMsg.Warning.warning_unpredictable_in_debug_state

Instruction is UNPREDICTABLE when executed in debug state.

ArchMsg.Warning.warning_unpredictable_mair_encoding

Fields:

ATTR unsigned int
: Memory attribute

ArchMsg.Warning.warning_unpredictable_pmu_counter_access

Access to PMU counters from non-secure EL0 or EL1 is UNPREDICTABLE with MDCR_EL2.HPMN set to 0.

ArchMsg.Warning.warning_unpredictable_prioritization_breakpoint_match_vector_catch

Two events with the same priority occurred at the same instruction: Address Matching Vector Catch debug event, and Breakpoint debug event It is constrained unpredictable which is taken.

ArchMsg.Warning.warning_unpredictable_stack_selection

UNPREDICTABLE state on context switch.

ArchMsg.Warning.warning_unpredictable_tsize_out_of_range

Fields:

TCR enum
: Which TCR

TSIZE signed int
: Value of TSize

TSIZE_MAX signed int
: The maximum allowed value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

TTBR unsigned int
Which TTBR region

ArchMsg.Warning.warning_unpredictable_unaligned_pc_as_base_register

The PC value must be word-aligned, otherwise the behavior of the instruction is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_unaligned_pop_stack

Fields:

FRAME_ADDR unsigned int
unaligned stack frame address

ArchMsg.Warning.warning_unpredictable_vmsa_memattrib

Fields:

TEXCB unsigned int
texcb value
TEX_REMAP bool
use tex remap

ArchMsg.Warning.warning_unpredictable_vtcr_t0sz_sl0

Fields:

SL0 signed int
VTCR.SL0, starting level for stage 2 translation table walks
T0SZ signed int
VTCR.T0SZ, required input address range

ArchMsg.Warning.warning_unpredictable_write_DHCSR

Fields:

NEW unsigned int
data attempted to be written
OLD unsigned int
previous value

ArchMsg.Warning.warning_unpredictable_write_SHCSR

UNPREDICTABLE setting a pending bit to 1 for an exception with priority >= execution priority.

ArchMsg.Warning.warning_unsupported_access_to_memory_mapped_register

Access to memory mapped register is unsupported.

ArchMsg.Warning.warning_user_jmcr_access

Fields:

WRITE unsigned int
access is write

ArchMsg.Warning.warning_wcr_mask_and_bas

DBGWCR_EL1.MASK is non-zero and BAS is not 0xff.

ArchMsg.Warning.warning_wcr_mask_reserved

DBGWCR_ELn.MASK is set to a reserved value.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_wcr_non_configuous_bas

DBGWCR_EL1.BAS has non-contiguous set of ones.

ArchMsg.Warning.warning_wvr_masked_bit_not_zero

DBGWVRn_EL1.VA contains a bit masked by DBGWCRn_EL1.MASK that is not zero.

ArchMsg.Why.why_branch_target_exception

A branch target exception has occurred. Fields:

BTYPE unsigned int
: pstate.BTYPE at time branch taken

OPCODE unsigned int
: opcode at branch target

PC unsigned int
: Virtual Address of branch target

ArchMsg.Why.why_illegal_state

An illegal mode change has occured. Fields:

MESSAGE string
: The message

BRANCH_MISPREDICT

Simulating branch mispredict. Fields:

PC unsigned int
: Origin address (or 0 if unavailable)

BRA_DIR

Direct branches, to immediate address. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR

Indirect branches, perhaps to a register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR_CCFAIL

Indirect branches fail, perhaps to a register.

BRA_RET_CCFAIL

Branche return fail, perhaps to a register.

CACHE_MAINTENANCE_OP

Cache Maintenance Operation. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Specified MVA or Set/way

FUNCTION enum
: Clean / Invalidate

SCOPE enum
: Affected region

SIDE enum
: Inst / Data.

CCFAIL

Conditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCFAIL_UNC

Unconditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS

Conditional instruction condition check pass. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS_UNC

Unconditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CHECKPOINT_MESSAGE

Report error messages from the checkpointing process. Fields:

message string
: Message contents.

CHECKPOINT_RESTORE_END

Checkpoint restore completed.

CHECKPOINT_RESTORE_START

Checkpoint restore about to start.

CHECKPOINT_SAVE_END

Checkpoint save completed.

CHECKPOINT_SAVE_START

Checkpoint save about to start.

CODE_CACHE_MAINT

Code cache maintenance. Fields:

END_INCL unsigned int
: Inclusive end addres for by VA or PA requests

NS bool
: Non-secure world for PA

START unsigned int
: Start address for by VA or PA requests

TYPE enum
: Request type.

COMPILE_BLOCK_END

Last instruction of basic block translated. Fields:

VADDR unsigned int
: Address of next instruction after this basic block

COMPILE_BLOCK_START

First instruction of basic block translated. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VADDR unsigned int
: Address of first instruction in basic block

COMPILE_INST

ARM instruction compiled. Fields:

DISASS string
: Disassembly of the instruction.

ISET enum
: The instruction set of this instruction.

ITSTATE unsigned int
: The ITSTATE current for the instruction.

OPCODE unsigned int
: The opcode of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

CONTEXTIDR

Context ID Register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

NS bool
: Secure or nonsecure banked register is accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

CONTEXT_SYNC

Called for every context synchronization event. Fields:

ARCH_CSE bool
: Architectural Context Synchronization Event

SYSREG_SYNC bool
: System register synchronization event

CORE_ENDIAN

Core BE8 Big-Endian state changed. Fields:

BE8 bool
: Core BE8 Big-Endian state

CORE_INFO

Static processor attributes. Only triggered by a call to DumpState(). Fields:

ARCH_PROFILE enum
: The architecture profile of the core.

CLUSTER_ID unsigned int
: The cluster ID of this processor.

CORE_NUM unsigned int
: The number of this core in an MP processor.

DETAILED_CORE_FEATURES enum
: Detailed core features as a bitfield. Enum values are bitmasks to extract the info from the field

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

FPU_VERSION enum
: The VFP version implemented by the core.

MEM_ARCH enum
: The memory architecture of the core.

MIDR_EL1 unsigned int
: MIDR_EL1 value

NUM_CORES unsigned int
: The number of cores in this MP processor.

QUANTUM_SIZE unsigned int
: The default quantum size of the core.

SECURITY_FEATURES bool
: Does the core have security features?

CORE_LOADS

Processor load accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_REGS

Changes of the core registers R0 to R14. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number, 0 to 14.

MODE enum
: Bank of the register accessed.

OLD_VALUE unsigned int
: The old value overwritten.

PHYS_ID enum
: The physical register accessed.

VALUE unsigned int
: The new value written to the register.

CORE_REGS64

Changes of the core registers X0..X30, SP_ELn. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: The register number, 0..30 for X0..X30, >=32 for SP_ELn.

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

CORE_STORES

Processor store accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CP14_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP14_WRITE

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

CP15_WRITE

System Control Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

CPSR

CPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

OLD_VALUE unsigned int
: The old CPSR value

UNKNOWN unsigned int
: Bits within the register that have unknown value.

VALUE unsigned int
: The new CPSR value

CRYPTO_SPEC

Every crypto instruction speculatively executed.

CorePowerStateChange

Core power-up state has changed. Fields:

PU bool
: Whether the core powered up

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DATA_CACHE_SET_ALLOC_TAGS

Set allocation tags from DC GVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of memory in bytes.

TAG unsigned int
: The value of the tag being set.

VADDR unsigned int
: Virtual address used for setting allocation tags.

DATA_CACHE_ZERO

Zero and invalidate cache line from DC ZVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of zero'd memory in bytes.

VADDR unsigned int
: Virtual address of the zero'd cache line.

DEBUG_EVENT

Hardware debug support event. Fields:

EVENT enum
: Description of event

VALUE unsigned int
: data value

END_COMPILE

Compilation end. Fields:

END_OF_PAGE_COUNT unsigned int
: Number of basic blocks exited due to page boundary since START_COMPILE.

FETCHFAIL_COUNT unsigned int
: Number of basic blocks exited due to ifetch failure START_COMPILE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: Number of instructions compiled since START_COMPILE.

NONSEQ_COUNT unsigned int
: Number of basic blocks exited due to non-sequential instructions since START_COMPILE.

PAGE_STRADDLE_COUNT unsigned int
: Number of basic blocks exited due to unaligned instructions crossing page since START_COMPILE.

ESB

Error Synchronization Barrier hit. Fields:

ASYNC bool
: ESB is async

IMPLICIT bool
: ESB is implicit

EXCEPTION

Exception taken. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

IS_PHYSICAL bool
: Physical or Virtual exception

LR unsigned int
: The value assigned to the link register.

PC unsigned int
: The location where the exception occurred.

PREFERRED_RETURN unsigned int
: The preferred return address for the exception.

TARGET_ISET enum
: The instruction set of the exception handler code.

TARGET_PC unsigned int
: The address the exception branches to.

VECTOR enum
: The exception vector.

VECTOR_OFFSET unsigned int
: The offset of an exception vector from the base address.

EXCEPTION_END

Every exception completed.

EXCEPTION_RAISE

Every exception raised.

EXCEPTION_RETURN

Branches on leaving exception. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

EXCEPTION_RETURN_PREBRANCH

Exception return event that must occur before PC and processor state have updated.

EXCEPTION_START

Every exception started.

FIQ_TAKEN

FIQ taken exception.

FP_STATE

Floating point state. Fields:

FPSR unsigned int
: FPSR register value.

MASK unsigned int
: The mask indicating updated cumulative bits.

HLT

HLT instruction occurred. Fields:

IMM16 unsigned int
: The 16-bit immediate value encoded in HLT instruction.

INST

Every instruction executed. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DEBUG_STATE bool
: The instruction is executed in debug state.

DISASS string
: Disassembly of instruction.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The current instruction set.

ITSTATE unsigned int
: The current ITSTATE.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

MODE enum
: The mode the core is in.

NS unsigned int
: The core's non-secure bit.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

OPCODE unsigned int
: The opcode of the instruction.

PADDR unsigned int
: The physical address of the instruction.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

INST_END

Every instruction completed.

INST_START

Every instruction started. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

INST_STRADDLE

Instruction straddles boundary. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

IRQ_TAKEN

IRQ taken exception.

ISB

ISB instruction occurred.

LOAD_MULTIPLE_REGS_LIST

List of destination registers (not source) of a multiple registers load instruction. Fields:

REG string
: Register to add into the list

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOCAL_MONITOR

Local monitor activity. Fields:

PADDR unsigned int
: Local Monitor Address

State enum
: State of the monitor (Open/Exclusive)

MEMMAP_DEBUG_READ

Memory mapped reads to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

MEMMAP_DEBUG_WRITE

Memory mapped writes to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

MMU_TRANS

Address translation information. Fields:

ASID unsigned int
: Address space identifier.

CORE_NUM unsigned int
: Core number in a multi processor.

Hyp bool
: Entry matches in Hyp state only

INNERCACHE_RA bool
: Is the inner cache allocate on read

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INNERCACHE_TRANSIENT bool
: Is the inner write-through transient

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

MEMTYPE enum
: Memory type.

NSDESC enum
: Is secure side supposed to access secure or nonsecure memory

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TRANSIENT bool
: Is the outer write-through transient

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

PADDR unsigned int
: Physical address of the access.

PAGESIZE unsigned int
: Page size as log2(size).

SH enum
: Shareability

SIDE enum
: Inst / Data.

STAGE1_PERM unsigned int
: Stage 1 permissions, [5:3] Privileged access (XWR) [2:0] User access (XWR)

STAGE2_PERM unsigned int
: Stage 2 permission mask [5:3] Privileged access (XWR) [2:0] User access (XWR)

VADDR unsigned int
: Virtual address of the access.

VMID unsigned int
: Virtual machine identifier.

XS bool
: XS Attribute

nG enum
: Flag indicating whether ASID will be matched

MMU_TTB_READ

This event is triggered by reads caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data read.

IPA unsigned int
: For stage 1, the IPA of the read.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OUTADDR unsigned int
: The output physical address.

PADDR unsigned int
: The physical address of the read.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MMU_TTB_WRITE

This event is triggered by writes caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data written.

IPA unsigned int
: For stage 1, the IPA of the write.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

PADDR unsigned int
: The physical address of the write.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MODE_CHANGE

Mode change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: The new mode.

NON_SECURE enum
: The new security state

OLD_MODE enum
: The old mode.

NOT_TAG_CHECKED_INST

MTE Tag not checked for current instruction.

PERIODIC

Called for every quantum. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PC unsigned int
The address of the next instruction to be executed on this CPU.

PMU_COUNTER_OVERFLOW

PMU counter overflow. Fields:

EVENT_ID unsigned int
Event Identifier
INDEX unsigned int
Counter index (as selected by PMSELR).
INTERRUPT bool
Is interrupt enabled on overflow for this counter

PREFETCH_MEMORY64

Prefetch from PRFM or PRFUM instructions. Fields:

PRFOP unsigned int
Prefetch hint
VADDR unsigned int
Virtual address of the location that should be prefetched.

PRELOAD_DATA

Data preload from PLD instruction. Fields:

VADDR unsigned int
Virtual address of the data that should be preloaded.

PRELOAD_INST

Instruction preload from PLI instruction. Fields:

VADDR unsigned int
Virtual address of the instruction that should be preloaded.

PRE_CORE_LOAD

Trace just before a core load. Fields:

CURRENT_TIME unsigned int
The core's current time, as simulated time plus local time.
LOCAL_TIME unsigned int
The core's local time, relative to the current quantum.

PRE_CORE_STORE

Trace just before a core store. Fields:

CURRENT_TIME unsigned int
The core's current time, as simulated time plus local time.
LOCAL_TIME unsigned int
The core's local time, relative to the current quantum.

PRE_TTB_READ

This event is triggered before reads caused by a translation table walk. Fields:

ADDR unsigned int
The physical address of the read.

PSTATE_PACTIVE_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions on PChannel. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

PSTATE_PACTIVE_OPMODE_TRANSITION_DENIED

Trace the denied pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_POWER_TRANSITION

Trace pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

PSTATE_PACTIVE_POWER_TRANSITION_DENIED

Trace the denied pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_THREAD_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions for threads on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

RUN_STATE

Run state transition. Fields:

INST_COUNT unsigned int
: Ticks count at point of transition.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NEW enum
: New run state.

OLD enum
: Old run state.

SEMIHOSTING_CALL

Call of a semihost function occurred. Fields:

PC unsigned int
: The program counter after the semihosting call.

SEMIHOSTING_PRECALL

About to call semihost. Fields:

EL2 bool
: Use EL2 translation regime.

NS bool
: Is Non-Secure.

REG_WIDTH unsigned int
: The current register width in bytes.

SIGNAL

External signal state change. Fields:

SIGNAL enum
: Signal that changed

STATE bool
: Signal asserted state

SOFTWARE_STEP

Return the Debug Software Step state. Fields:

STATE enum
: Software Step state

SPSR

SPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: Which of the banked SPSR registers is written.

OLD_VALUE unsigned int
: The old SPSR value

VALUE unsigned int
: The new SPSR value

START_COMPILE

Compilation started. Fields:

VADDR unsigned int
: Instruction where compilation begins.

STG_MULTIPLE_MICRO_OPS_ADDRESS_RANGE

Address range for MTE stores with multiple uops. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

STORE_MULTIPLE_REGS_ADDRESS_RANGE

Address range for stores with multiple registers. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

SUBOPTIMAL_LDST_RETIRED

suboptimal load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1: half-word, 2: word ...)

OPTIMAL_ALIGN_SELECTOR unsigned int
: Selector to decide the optimal alignment size (1: access width, 2: 4-bytes, 3: 8-bytes, ..., 12: 4-kbytes)

VADDR unsigned int
: The virtual address of the access

SYNC

Called for every synchronization, that is every quantum break. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

LOCAL_QUANTUM unsigned int
: The local quantum of this CPU.

LOCAL_TIME unsigned int
: The local time of this CPU.

SYNC_REASON enum
: Reason for sync.

SYSCALL

System call instruction executed. Fields:

IMM unsigned int
: Immediate value of the system call instruction.

TYPE enum
: System call type.

VADDR unsigned int
: Instruction that caused the system call.

SYSREG_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE32

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE64

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

UNALIGNED_LDST_RETIRED

Processor unaligned load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1=>half-word, 2->word ...)

NUMBER_OF_BEATS unsigned int
: Number of accesses (beats) in this burst (Total size in bytes is ACCESS_SIZE * NUMBER_OF_BEATS)

VADDR unsigned int
: The virtual address of the access

VFP_D_REGS

VFP/NEON D 64 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_Q_REGS

VFP/NEON Q 128 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding byte in the register. For example, 0x1 would indicate a write to VALUE[7:0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_SYS_REGS

Writes to the VFP/NEON units system registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: Which VFP system register is written.

MASK unsigned int
: The mask indicating updated cumulative bits.

OLD_VALUE unsigned int
: The register's old value overwritten.

VALUE unsigned int
: The new value written to the VFP system register.

VFP_S_REGS

VFP/NEON S 32 bit register write. Fields:

ALIASING enum
: Type of register aliasing used.

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

WAYPOINT

Signals end of basic block execution. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: Origin instructions set

IS_COND bool
: Indicates if this is a conditional waypoint.

PC unsigned int
: Origin address (or 0 if unavailable)

TAKEN bool
: Indicates if this waypoint was taken.

TARGET unsigned int
: Destination address

TARGET_ISET enum
: Destination instructions set

WFE_END

WFE ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFE.

WFE_EVENT_REGISTER

WFE event register status: set/clear, reason. Fields:

INST_COUNT unsigned int
: Ticks count.

REASON enum
: Reason for set/clear. Only REASON==0 (Cleared by WFE) clears the bit, all other reasons set it.

WFE_IGNORED

WFE ignored. Fields:

EVENT bool
: This WFE was ignored because the event register was set.

INST_COUNT unsigned int
: Ticks count when ignoring WFE.

TRAPPED bool
: This WFE was trapped.

WFE_START

WFE entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFE.

WFI_END

WFI ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFI.

WFI_IGNORED

WFI ignored. Fields:

DISABLED bool
: This WFI was ignored because WFI is disabled.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: Ticks count when ignoring WFI.

REASON enum
: specifies reason why WFI trace was ignored

TRAPPED bool
: This WFI was trapped.

WFI_START

WFI entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFI.

WFI_WAKEUP

WFI wakeup. Fields:

INST_COUNT unsigned int
: Ticks count when WFI wakeup occurred.

REASON enum
: Reason for wakeup.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.20 ARM_Cortex-A75 - trace

This section describes the trace sources.

ASYNC_MEMORY_FAULT

Context ID Register write. Fields:

FAULT unsigned int
: Fault status in ESR format

PADDR unsigned int
: Physical Address (or 0 if unavailable)

VADDR signed int
: Virtual Address (or 0 if unavailable)

ATOMIC_END_ACCESS

Bus trace access for atomics. Fields:

ACCESS_FAIL bool
: Memory access failed

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

LOAD_VALUE unsigned int
: Loaded values

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ATOMIC_START_ACCESS

Bus trace access for atomics. Fields:

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

NSDESC unsigned int
: The security state of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ArchMsg.Error.exit_code

Fields:

EXITCODE signed int
: Exit code to be returned

KIND string
: Component kind that invoked the exit code trace

REASON string
: Reason why the exit code trace was invoked (optional)

ArchMsg.Warning.IgnoredWarmResetRequest

Event fires when software attempts to request warm reset through write to deprecated DBGPRCR.CWRR.

ArchMsg.Warning.RAS_ErrselrUnpredictableState

ERRSELR is in unpredictable state due to value greater than number_of_error_records written to it.

ArchMsg.Warning.RAS_UnpredictableStateWrite

Fields:

REG_NAME string
: Name of register being written.

ArchMsg.Warning.TarmacReconstructor

Unexpected event happened during Tarmac reconstruction. Fields:

REASON string
: Reason

ArchMsg.Warning.Unpredictably Indexed PM Event Register

Fields:

IsDirect unsigned int
: Direct Access rather than indirect

N unsigned int
: Number of Accessible Registers

SEL unsigned int
: Selector

ArchMsg.Warning.branch_to_unaligned_address

Fields:

ADDR unsigned int
: Unaligned branch address

ArchMsg.Warning.cache_contents_unknown

A cache has been activated without being invalidated and may contain UNKNOWN content. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

REGIME unsigned int
: Current translation regime

SIDE unsigned int
: 0 d-side cache or unified cache, 1 i-side

ArchMsg.Warning.dcimvac_matches_watchpoint

Watchpoints matching an AArch32 DCIMVA are implementation defined.

ArchMsg.Warning.decode_fieldmismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_fracbitsnegative

VCVT Instruction is unpredictable with a negative number of fraction bits.

ArchMsg.Warning.decode_initblock

Instruction is unpredictable in an IT block.

ArchMsg.Warning.decode_initblocknotlast

Instruction is unpredictable in an IT block when not the last.

ArchMsg.Warning.decode_invalid_condition

Instruction is unpredictable when the condition is not AL.

ArchMsg.Warning.decode_invalidfieldcombination

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_invalidvalue

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_mem_hint_unallocated

There is no memory hint allocated to this bit pattern.

ArchMsg.Warning.decode_registermatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_registermismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registeroutofrange

Float point instruction is unpredictable indexing beyond the end of the register bank.

ArchMsg.Warning.decode_sbzsbo

Reserved bits in the instruction are not canonical.

ArchMsg.Warning.decode_transactiontoobig

Attempt to perform a memory transaction of more than 128 bytes.

ArchMsg.Warning.decode_unpred_other

Instruction is unpredictable.

ArchMsg.Warning.decode_unpreduseofpc

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13inreglist

Instruction is unpredictable with R13 in register-list.

ArchMsg.Warning.decode_useofd16orhigher

Attempt to use register D16 or higher where it is not allowed.

ArchMsg.Warning.decode_writebackandbaseinlist

Instruction is unpredictable with write back when the base register is in the transfer list.

ArchMsg.Warning.decode_zeroregistersinlist

Instruction is unpredictable when no registers are to be transfered.

ArchMsg.Warning.recursive_branch

An instruction is performing a branch that targets the same instruction. If this was intended then a WFI might be more effective. Fields:

PC unsigned int
: Address of instruction causing the branch

ArchMsg.Warning.recursive_exception

An instruction has generated an exception that targets the same instruction. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CPSR unsigned int
: Processor state

PC unsigned int
: Address of instruction causing the exception

TYPE enum
: Type of exception

ArchMsg.Warning.reentrant_vector_catch

Fields:

ADDR unsigned int
: vector address

ArchMsg.Warning.reserved_it_state

Fields:

ITSTATE unsigned int
: reserved IT state

ArchMsg.Warning.tlb_contents_unknown

Translation has been enabled without invalidating the TLB. Fields:

EL unsigned int
: Exception level of translation regime

INVALIDITY unsigned int
: Which TLB may be invalid (if TLBs are separate)

NON_SECURE unsigned int
: Security state of translation regime

VMID unsigned int
: Current VMID, if applicable

ArchMsg.Warning.unknown_DLR_DSPSR

Exiting debug state with unknown DLR or DSPSR.

ArchMsg.Warning.unknown_ELR_SPSR

Returning from debug exception with unknown ELR or SPSR.

ArchMsg.Warning.unpredictable_a32_breakpoint

Fields:

ADDR1 unsigned int
: Address

ADDR2 unsigned int
: Breakpoint address

BAS unsigned int
: BAS Field

IS_ADDRESS_MISMATCH_BREAKPOINT bool
: Is Mismatch breakpoint, not a match breakpoint

N unsigned int
: Breakpoint number

OFFSET unsigned int
: Indicates which halfword actually considering into the A32 instruction

ArchMsg.Warning.unpredictable_banked_register_access

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

R unsigned int
R bit
SYSM unsigned int
SYSm

ArchMsg.Warning.unpredictable_ccfail_undef

Fields:

OPCODE unsigned int
opcode of the instruction
VADDR unsigned int
Virtual address of the instruction being executed

ArchMsg.Warning.unpredictable_ici_bits

ICI bits do not apply or are architecturally unpredictable for instruction.

ArchMsg.Warning.unpredictable_t32_breakpoint

Fields:

ADDR1 unsigned int
Address
ADDR2 unsigned int
Breakpoint address
BAS unsigned int
BAS Field
IS_ADDRESS_MISMATCH_BREAKPOINT bool
Is Mismatch breakpoint, not a match breakpoint
N unsigned int
Breakpoint number
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_t32_vector_catch

Fields:

ADDR unsigned int
Address
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_watchpoint_far

Fields:

LOWER_BOUND unsigned int
the lowest address accessed by the instruction that triggered the watchpoint
UPPER_BOUND unsigned int
the highest watchpointed address accessed by that instruction

ArchMsg.Warning.warning_AdvSIMDExpandImmUnexpectedZero

AdvSIMDExpandImm may treat this immediate value as UNPREDICTABLE.

ArchMsg.Warning.warning_ConditionalSMC

SMC instruction has UNPREDICTABLE effects when conditional when combined with traps.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_access_crosses_page_boundary

Access crosses page boundary.

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault_on_both_pages

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_spanning_different_memory

Fields:

ADDR unsigned int
: address of access crossing page

MEMTYPE_PAGE1 signed int
: First Page's memory type (0-Normal, 1-Non-Normal)

MEMTYPE_PAGE2 signed int
: Second Page's memory type (0-Normal, 1-Non-Normal)

ArchMsg.Warning.warning_access_wraps_around_memory

Fields:

WIDTH unsigned int
: Address width

ArchMsg.Warning.warning_bx_from_thumbee

Fields:

ADDR unsigned int
: Destination address

ArchMsg.Warning.warning_ccsidr_unimplemented_level

Fields:

CSSELR unsigned int
: current effective value of CSSELR

ArchMsg.Warning.warning_change_to_ns_when_tge_set

Attempting to change NS to 1 when HCR.TGE = 1.

ArchMsg.Warning.warning_contiguous_bit_check_abort

Fields:

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ArchMsg.Warning.warning_contiguous_bit_error

Fields:

CONTIG_ADDR unsigned int
: address of the TLB first read, with which this entry is expected to be contiguous

CONTIG_DATA unsigned int
: expected contents based on the entry at CONTIG_ADDR

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ENTRY_DATA unsigned int
: contents of conflicting TLB entry

ArchMsg.Warning.warning_cp10_cp11_mismatch

Fields:

EL unsigned int
: EL owning the control

ArchMsg.Warning.warning_cp10_cp11_reserved_value

Fields:

CP10 unsigned int
: CP10 access

CP11 unsigned int
: CP11 access

ArchMsg.Warning.warning_csselr_level_out_of_range

Fields:

IMPLEMENTED unsigned int
: number of cache levels implemented

LEVEL unsigned int
: written value of CSSELR.Level

TYPE enum
: The type of cache selected by CSSELR.InD and TnD

ArchMsg.Warning.warning_debug_flow_control_bits_not_obeyed

Fields:

ERROR enum
: type of error resulting

REGISTER enum
: register under consideration

ArchMsg.Warning.warning_debug_register_access_during_reset

Fields:

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

ArchMsg.Warning.warning_decode_cps_inconsistent_fields

The mode setting fields in the CPS instruction are inconsistent.

ArchMsg.Warning.warning_decode_invalid_state

Instruction is unpredictable in current exception-level/security state.

ArchMsg.Warning.warning_default_cacheable_mmu_on

Default cacheable is enabled (HCR.DC = 1) while MMU is enabled.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_default_cacheable_vmmu_off

Default cacheable is enabled (HCR.DC = 1) while HCR.VM = 0.

ArchMsg.Warning.warning_deprecated_wvr_bit_2_set

Fields:

N unsigned int
: index

WVR unsigned int
: DBGWVR

ArchMsg.Warning.warning_eret_while_software_step_active_pending

Missing ISB between setting MDSCR_EL1.SS and ERET.

ArchMsg.Warning.warning_exclusive_to_non_normal

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_exclusive_to_non_writeback

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_execute_from_device_memory

An attempt was made to execute from Device memory.

ArchMsg.Warning.warning_illegal_cpsr_mode

Fields:

NEW_MODE unsigned int
: New value of CPSR.M

ArchMsg.Warning.warning_illegal_srs_mode

Fields:

MODE unsigned int
: mode for Banked SP

ArchMsg.Warning.warning_implementation_defined_read_debug_register_in_SCS

Read Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_implementation_defined_sequential_security_transitions_supported

The behaviour of a sequential instruction fetches that cross from non-secure to secure memory and contain SG instruction is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_implementation_defined_stack_limit_check_supported

It is IMPLEMENTATION DEFINED whether stack pointer limit checking is performed for the this instructions. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDRESS unsigned int
: Address of instruction

OPCODE unsigned int
: instruction opcode

ArchMsg.Warning.warning_implementation_defined_write_debug_register_in_SCS

Write Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

DATA unsigned int
: data attempted to be written

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_invalid_tcr_granule

Fields:

REQUEST signed int
: page size requested (or 0 for reserved)

SUBSTITUTE signed int
: best guess available page size

TG_ID bool
: bits TG0 or TG1

ArchMsg.Warning.warning_load_multiple_user_registers_from_user_mode

An LDM(user registers) instruction executed from user mode.

ArchMsg.Warning.warning_load_pc_from_unaligned

PC loaded from an unaligned location.

ArchMsg.Warning.warning_mte_in_mem_attr_not_supported

Fields:

ATTR unsigned int
: Memory attribute

ArchMsg.Warning.warning_reserved_breakpoint_state_match

Fields:

IS_BREAKPOINT unsigned int
: Is breakpoint, not watchpoint

ArchMsg.Warning.warning_shareability

Unpredictable: combination of the 1st and 2nd stages of translation is Normal Inner Non-Cacheable, Outer Non-Cacheable.

ArchMsg.Warning.warning_software_step_set_while_enabled

MDSCR_EL1.SS set to 1 while software step debug exceptions are enabled.

ArchMsg.Warning.warning_thumb_instruction_wraps_around_memory

Thumb instruction warps around memory.

ArchMsg.Warning.warning_ttbr_sbz_bits_are_not_zero

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MSB unsigned int
x-1
TTBR enum
which TTBR register

ArchMsg.Warning.warning_unaligned_address_dbgdtrrx_write

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_address_dbgdtrtx_read

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_device

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_strongly_ordered

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unknown_sau_rnr

SAU_RNR was set to an unsupported value.

ArchMsg.Warning.warning_unpredictable_AIRCR_PRIP_and_BFHFNMINP

The effect of setting both AIRCR.BFHFNMINP and AIRCR.PRIP to 1 is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTCLRACTIVE_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTCLRACTIVE if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_and_SYSRESETREQ

When the processor is halted in Debug state, if a write to the register writes a 1 to both VECTRESET and SYSRESETREQ, the behavior is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTRESET if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_incorrect_VKEY

The value 0x05FA must be written to AIRCR.VKEY, otherwise the register write is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_EXE_RETURN_Reserved_Bit

EXC_RETURN[23:7] are reserved with the special condition that all bits should be written as one. Values other than all 1s are UNPREDICTABLE. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

EXC_RETURN unsigned int
Special PC value
SBO_MASK unsigned int
Which bits Should Be One to not UNPRED

ArchMsg.Warning.warning_unpredictable_change_to_DEMCR_MON_STEP_at_insufficient_priority

The effect of changing DEMCR.MON_STEP at an execution priority that is lower than the priority of the DebugMonitor exception is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_change_to_priority_of_active_exception

Changing the priority of an active exception is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_clear_lspact

Fields:

DRVEXC_TAKEN unsigned int
derived exception taken not pended
REACHED_LAST_STORE unsigned int
only the last faulted

ArchMsg.Warning.warning_unpredictable_dbg_exit_unaligned_dlr

Debug exit to AArch32 with DLR[0] set to 1. PC[0] can be set to 0 or to DLR[0].

ArchMsg.Warning.warning_unpredictable_exception_catch

Generation of an exception catch event is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_inconsistent_state

state on exception return is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_instruction

exception return instruction is unpredictable.

ArchMsg.Warning.warning_unpredictable_in_debug_state

Instruction is UNPREDICTABLE when executed in debug state.

ArchMsg.Warning.warning_unpredictable_mair_encoding

Fields:

ATTR unsigned int
Memory attribute

ArchMsg.Warning.warning_unpredictable_pmu_counter_access

Access to PMU counters from non-secure EL0 or EL1 is UNPREDICTABLE with MDCR_EL2.HPMN set to 0.

ArchMsg.Warning.warning_unpredictable_prioritization_breakpoint_match_vector_catch

Two events with the same priority occurred at the same instruction: Address Matching Vector Catch debug event, and Breakpoint debug event It is constrained unpredictable which is taken.

ArchMsg.Warning.warning_unpredictable_stack_selection

UNPREDICTABLE state on context switch.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_tsize_out_of_range

Fields:

TCR enum
: Which TCR

TSIZE signed int
: Value of TSize

TSIZE_MAX signed int
: The maximum allowed value

TTBR unsigned int
: Which TTBR region

ArchMsg.Warning.warning_unpredictable_unaligned_pc_as_base_register

The PC value must be word-aligned, otherwise the behavior of the instruction is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_unaligned_pop_stack

Fields:

FRAME_ADDR unsigned int
: unaligned stack frame address

ArchMsg.Warning.warning_unpredictable_vmsa_memattrib

Fields:

TEXCB unsigned int
: texcb value

TEX_REMAP bool
: use tex remap

ArchMsg.Warning.warning_unpredictable_vtcr_t0sz_sl0

Fields:

SL0 signed int
: VTCR.SL0, starting level for stage 2 translation table walks

T0SZ signed int
: VTCR.T0SZ, required input address range

ArchMsg.Warning.warning_unpredictable_write_DHCSR

Fields:

NEW unsigned int
: data attempted to be written

OLD unsigned int
: previous value

ArchMsg.Warning.warning_unpredictable_write_SHCSR

UNPREDICTABLE setting a pending bit to 1 for an exception with priority >= execution priority.

ArchMsg.Warning.warning_unsupported_access_to_memory_mapped_register

Access to memory mapped register is unsupported.

ArchMsg.Warning.warning_user_jmcr_access

Fields:

WRITE unsigned int
: access is write

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_wcr_mask_and_bas

DBGWCR_EL1.MASK is non-zero and BAS is not 0xff.

ArchMsg.Warning.warning_wcr_mask_reserved

DBGWCR_ELn.MASK is set to a reserved value.

ArchMsg.Warning.warning_wcr_non_configuous_bas

DBGWCR_EL1.BAS has non-contiguous set of ones.

ArchMsg.Warning.warning_wvr_masked_bit_not_zero

DBGWVRn_EL1.VA contains a bit masked by DBGWCRn_EL1.MASK that is not zero.

ArchMsg.Why.why_branch_target_exception

A branch target exception has occurred. Fields:

BTYPE unsigned int
: pstate.BTYPE at time branch taken

OPCODE unsigned int
: opcode at branch target

PC unsigned int
: Virtual Address of branch target

ArchMsg.Why.why_illegal_state

An illegal mode change has occured. Fields:

MESSAGE string
: The message

BRANCH_MISPREDICT

Simulating branch mispredict. Fields:

PC unsigned int
: Origin address (or 0 if unavailable)

BRA_DIR

Direct branches, to immediate address. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

BRA_INDIR

Indirect branches, perhaps to a register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR_CCFAIL

Indirect branches fail, perhaps to a register.

BRA_RET_CCFAIL

Branche return fail, perhaps to a register.

CACHE_MAINTENANCE_OP

Cache Maintenance Operation. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Specified MVA or Set/way

FUNCTION enum
: Clean / Invalidate

SCOPE enum
: Affected region

SIDE enum
: Inst / Data.

CCFAIL

Conditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCFAIL_UNC

Unconditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS

Conditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS_UNC

Unconditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CHECKPOINT_MESSAGE

Report error messages from the checkpointing process. Fields:

message string
: Message contents.

CHECKPOINT_RESTORE_END

Checkpoint restore completed.

CHECKPOINT_RESTORE_START

Checkpoint restore about to start.

CHECKPOINT_SAVE_END

Checkpoint save completed.

CHECKPOINT_SAVE_START

Checkpoint save about to start.

CODE_CACHE_MAINT

Code cache maintenance. Fields:

END_INCL unsigned int
: Inclusive end addres for by VA or PA requests

NS bool
: Non-secure world for PA

START unsigned int
: Start address for by VA or PA requests

TYPE enum
: Request type.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

COMPILE_BLOCK_END

Last instruction of basic block translated. Fields:

VADDR unsigned int
: Address of next instruction after this basic block

COMPILE_BLOCK_START

First instruction of basic block translated. Fields:

VADDR unsigned int
: Address of first instruction in basic block

COMPILE_INST

ARM instruction compiled. Fields:

DISASS string
: Disassembly of the instruction.

ISET enum
: The instruction set of this instruction.

ITSTATE unsigned int
: The ITSTATE current for the instruction.

OPCODE unsigned int
: The opcode of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

CONTEXTIDR

Context ID Register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

NS bool
: Secure or nonsecure banked register is accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

CONTEXT_SYNC

Called for every context synchronization event. Fields:

ARCH_CSE bool
: Architectural Context Synchronization Event

SYSREG_SYNC bool
: System register synchronization event

CORE_ENDIAN

Core BE8 Big-Endian state changed. Fields:

BE8 bool
: Core BE8 Big-Endian state

CORE_INFO

Static processor attributes. Only triggered by a call to DumpState(). Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ARCH_PROFILE enum
: The architecture profile of the core.

CLUSTER_ID unsigned int
: The cluster ID of this processor.

CORE_NUM unsigned int
: The number of this core in an MP processor.

DETAILED_CORE_FEATURES enum
: Detailed core features as a bitfield. Enum values are bitmasks to extract the info from the field

FPU_VERSION enum
: The VFP version implemented by the core.

MEM_ARCH enum
: The memory architecture of the core.

MIDR_EL1 unsigned int
: MIDR_EL1 value

NUM_CORES unsigned int
: The number of cores in this MP processor.

QUANTUM_SIZE unsigned int
: The default quantum size of the core.

SECURITY_FEATURES bool
: Does the core have security features?

CORE_LOADS

Processor load accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CORE_REGS

Changes of the core registers R0 to R14. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number, 0 to 14.

MODE enum
: Bank of the register accessed.

OLD_VALUE unsigned int
: The old value overwritten.

PHYS_ID enum
: The physical register accessed.

VALUE unsigned int
: The new value written to the register.

CORE_REGS64

Changes of the core registers X0..X30, SP_ELn. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: The register number, 0..30 for X0..X30, >=32 for SP_ELn.

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

CORE_STORES

Processor store accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CP14_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP14_WRITE

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CP15_WRITE

System Control Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

CPSR

CPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

OLD_VALUE unsigned int
: The old CPSR value

UNKNOWN unsigned int
: Bits within the register that have unknown value.

VALUE unsigned int
: The new CPSR value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CRYPTO_SPEC

Every crypto instruction speculatively executed.

CorePowerStateChange

Core power-up state has changed. Fields:

PU bool
: Whether the core powered up

DATA_CACHE_SET_ALLOC_TAGS

Set allocation tags from DC GVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of memory in bytes.

TAG unsigned int
: The value of the tag being set.

VADDR unsigned int
: Virtual address used for setting allocation tags.

DATA_CACHE_ZERO

Zero and invalidate cache line from DC ZVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of zero'd memory in bytes.

VADDR unsigned int
: Virtual address of the zero'd cache line.

DEBUG_EVENT

Hardware debug support event. Fields:

EVENT enum
: Description of event

VALUE unsigned int
: data value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

END_COMPILE

Compilation end. Fields:

END_OF_PAGE_COUNT unsigned int
: Number of basic blocks exited due to page boundary since START_COMPILE.

FETCHFAIL_COUNT unsigned int
: Number of basic blocks exited due to ifetch failure START_COMPILE.

INST_COUNT unsigned int
: Number of instructions compiled since START_COMPILE.

NONSEQ_COUNT unsigned int
: Number of basic blocks exited due to non-sequential instructions since START_COMPILE.

PAGE_STRADDLE_COUNT unsigned int
: Number of basic blocks exited due to unaligned instructions crossing page since START_COMPILE.

ESB

Error Synchronization Barrier hit. Fields:

ASYNC bool
: ESB is async

IMPLICIT bool
: ESB is implicit

EXCEPTION

Exception taken. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

IS_PHYSICAL bool
: Physical or Virtual exception

LR unsigned int
: The value assigned to the link register.

PC unsigned int
: The location where the exception occurred.

PREFERRED_RETURN unsigned int
: The preferred return address for the exception.

TARGET_ISET enum
: The instruction set of the exception handler code.

TARGET_PC unsigned int
: The address the exception branches to.

VECTOR enum
: The exception vector.

VECTOR_OFFSET unsigned int
: The offset of an exception vector from the base address.

EXCEPTION_END

Every exception completed.

EXCEPTION_RAISE

Every exception raised.

EXCEPTION_RETURN

Branches on leaving exception. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

EXCEPTION_RETURN_PREBRANCH

Exception return event that must occur before PC and processor state have updated.

EXCEPTION_START

Every exception started.

FIQ_TAKEN

FIQ taken exception.

FP_STATE

Floating point state. Fields:

FPSR unsigned int
: FPSR register value.

MASK unsigned int
: The mask indicating updated cumulative bits.

HLT

HLT instruction occurred. Fields:

IMM16 unsigned int
: The 16-bit immediate value encoded in HLT instruction.

INST

Every instruction executed. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DEBUG_STATE bool
: The instruction is executed in debug state.

DISASS string
: Disassembly of instruction.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The current instruction set.

ITSTATE unsigned int
: The current ITSTATE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

MODE enum
: The mode the core is in.

NS unsigned int
: The core's non-secure bit.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

OPCODE unsigned int
: The opcode of the instruction.

PADDR unsigned int
: The physical address of the instruction.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

INST_END

Every instruction completed.

INST_START

Every instruction started. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

INST_STRADDLE

Instruction straddles boundary. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

IRQ_TAKEN

IRQ taken exception.

ISB

ISB instruction occurred.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOAD_MULTIPLE_REGS_LIST

List of destination registers (not source) of a multiple registers load instruction. Fields:

REG string
: Register to add into the list

LOCAL_MONITOR

Local monitor activity. Fields:

PADDR unsigned int
: Local Monitor Address

State enum
: State of the monitor (Open/Exclusive)

MEMMAP_DEBUG_READ

Memory mapped reads to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

MEMMAP_DEBUG_WRITE

Memory mapped writes to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

MMU_TRANS

Address translation information. Fields:

ASID unsigned int
: Address space identifier.

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Hyp bool
: Entry matches in Hyp state only

INNERCACHE_RA bool
: Is the inner cache allocate on read

INNERCACHE_TRANSIENT bool
: Is the inner write-through transient

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

MEMTYPE enum
: Memory type.

NSDESC enum
: Is secure side supposed to access secure or nonsecure memory

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TRANSIENT bool
: Is the outer write-through transient

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

PADDR unsigned int
: Physical address of the access.

PAGESIZE unsigned int
: Page size as log2(size).

SH enum
: Shareability

SIDE enum
: Inst / Data.

STAGE1_PERM unsigned int
: Stage 1 permissions, [5:3] Privileged access (XWR) [2:0] User access (XWR)

STAGE2_PERM unsigned int
: Stage 2 permission mask [5:3] Privileged access (XWR) [2:0] User access (XWR)

VADDR unsigned int
: Virtual address of the access.

VMID unsigned int
: Virtual machine identifier.

XS bool
: XS Attribute

nG enum
: Flag indicating whether ASID will be matched

MMU_TTB_READ

This event is triggered by reads caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data read.

IPA unsigned int
: For stage 1, the IPA of the read.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

OUTADDR unsigned int
: The output physical address.

PADDR unsigned int
: The physical address of the read.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MMU_TTB_WRITE

This event is triggered by writes caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data written.

IPA unsigned int
: For stage 1, the IPA of the write.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

PADDR unsigned int
: The physical address of the write.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MODE_CHANGE

Mode change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: The new mode.

NON_SECURE enum
: The new security state

OLD_MODE enum
: The old mode.

NOT_TAG_CHECKED_INST

MTE Tag not checked for current instruction.

PERIODIC

Called for every quantum. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

PC unsigned int
: The address of the next instruction to be executed on this CPU.

PMU_COUNTER_OVERFLOW

PMU counter overflow. Fields:

EVENT_ID unsigned int
: Event Identifier

INDEX unsigned int
: Counter index (as selected by PMSELR).

INTERRUPT bool
: Is interrupt enabled on overflow for this counter

PREFETCH_MEMORY64

Prefetch from PRFM or PRFUM instructions. Fields:

PRFOP unsigned int
: Prefetch hint

VADDR unsigned int
: Virtual address of the location that should be prefetched.

PRELOAD_DATA

Data preload from PLD instruction. Fields:

VADDR unsigned int
: Virtual address of the data that should be preloaded.

PRELOAD_INST

Instruction preload from PLI instruction. Fields:

VADDR unsigned int
: Virtual address of the instruction that should be preloaded.

PRE_CORE_LOAD

Trace just before a core load. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_CORE_STORE

Trace just before a core store. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_TTB_READ

This event is triggered before reads caused by a translation table walk. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR unsigned int
The physical address of the read.

PSTATE_PACTIVE_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Operating mode after the transition
OLD_STATE enum
Operating mode before the transition

PSTATE_PACTIVE_OPMODE_TRANSITION_DENIED

Trace the denied pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Operating mode after the transition
OLD_STATE enum
Operating mode before the transition
REASON string
Reason for the transition being denied

PSTATE_PACTIVE_POWER_TRANSITION

Trace pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Power mode after the transition
OLD_STATE enum
Power mode before the transition

PSTATE_PACTIVE_POWER_TRANSITION_DENIED

Trace the denied pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Power mode after the transition
OLD_STATE enum
Power mode before the transition
REASON string
Reason for the transition being denied

PSTATE_PACTIVE_THREAD_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions for threads on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Operating mode after the transition
OLD_STATE enum
Operating mode before the transition

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

RUN_STATE

Run state transition. Fields:

INST_COUNT unsigned int
: Ticks count at point of transition.

NEW enum
: New run state.

OLD enum
: Old run state.

SEMIHOSTING_CALL

Call of a semihost function occurred. Fields:

PC unsigned int
: The program counter after the semihosting call.

SEMIHOSTING_PRECALL

About to call semihost. Fields:

EL2 bool
: Use EL2 translation regime.

NS bool
: Is Non-Secure.

REG_WIDTH unsigned int
: The current register width in bytes.

SIGNAL

External signal state change. Fields:

SIGNAL enum
: Signal that changed

STATE bool
: Signal asserted state

SOFTWARE_STEP

Return the Debug Software Step state. Fields:

STATE enum
: Software Step state

SPSR

SPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: Which of the banked SPSR registers is written.

OLD_VALUE unsigned int
: The old SPSR value

VALUE unsigned int
: The new SPSR value

START_COMPILE

Compilation started. Fields:

VADDR unsigned int
: Instruction where compilation begins.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

STG_MULTIPLE_MICRO_OPS_ADDRESS_RANGE

Address range for MTE stores with multiple uops. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

STORE_MULTIPLE_REGS_ADDRESS_RANGE

Address range for stores with multiple registers. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

SUBOPTIMAL_LDST_RETIRED

suboptimal load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1: half-word, 2: word ...)

OPTIMAL_ALIGN_SELECTOR unsigned int
: Selector to decide the optimal alignment size (1: access width, 2: 4-bytes, 3: 8-bytes, ..., 12: 4-kbytes)

VADDR unsigned int
: The virtual address of the access

SYNC

Called for every synchronization, that is every quantum break. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

LOCAL_QUANTUM unsigned int
: The local quantum of this CPU.

LOCAL_TIME unsigned int
: The local time of this CPU.

SYNC_REASON enum
: Reason for sync.

SYSCALL

System call instruction executed. Fields:

IMM unsigned int
: Immediate value of the system call instruction.

TYPE enum
: System call type.

VADDR unsigned int
: Instruction that caused the system call.

SYSREG_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE32

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE64

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

UNALIGNED_LDST_RETIRED

Processor unaligned load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1=>half-word, 2->word ...)

NUMBER_OF_BEATS unsigned int
: Number of accesses (beats) in this burst (Total size in bytes is ACCESS_SIZE * NUMBER_OF_BEATS)

VADDR unsigned int
: The virtual address of the access

VFP_D_REGS

VFP/NEON D 64 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ID unsigned int
The register number.
MASK unsigned int
Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]
OLD_VALUE unsigned int
The old value overwritten.
VALUE unsigned int
The new value written to the register.

VFP_Q_REGS

VFP/NEON Q 128 bit register write. Fields:

ALIASING enum
Type of register aliasing used
CORE_NUM unsigned int
Core number in a multi processor.
ID unsigned int
The register number.
MASK unsigned int
Each bit indicates a write to the corresponding byte in the register. For example, 0x1 would indicate a write to VALUE[7:0]
OLD_VALUE unsigned int
The old value overwritten.
VALUE unsigned int
The new value written to the register.

VFP_SYS_REGS

Writes to the VFP/NEON units system registers. Fields:

CORE_NUM unsigned int
Core number in a multi processor.
ID enum
Which VFP system register is written.
MASK unsigned int
The mask indicating updated cumulative bits.
OLD_VALUE unsigned int
The register's old value overwritten.
VALUE unsigned int
The new value written to the VFP system register.

VFP_S_REGS

VFP/NEON S 32 bit register write. Fields:

ALIASING enum
Type of register aliasing used.
CORE_NUM unsigned int
Core number in a multi processor.
ID unsigned int
The register number.
MASK unsigned int
Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]
OLD_VALUE unsigned int
The old value overwritten.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VALUE unsigned int
: The new value written to the register.

WAYPOINT

Signals end of basic block execution. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: Origin instructions set

IS_COND bool
: Indicates if this is a conditional waypoint.

PC unsigned int
: Origin address (or 0 if unavailable)

TAKEN bool
: Indicates if this waypoint was taken.

TARGET unsigned int
: Destination address

TARGET_ISET enum
: Destination instructions set

WFE_END

WFE ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFE.

WFE_EVENT_REGISTER

WFE event register status: set/clear, reason. Fields:

INST_COUNT unsigned int
: Ticks count.

REASON enum
: Reason for set/clear. Only REASON==0 (Cleared by WFE) clears the bit, all other reasons set it.

WFE_IGNORED

WFE ignored. Fields:

EVENT bool
: This WFE was ignored because the event register was set.

INST_COUNT unsigned int
: Ticks count when ignoring WFE.

TRAPPED bool
: This WFE was trapped.

WFE_START

WFE entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFE.

WFI_END

WFI ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFI.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

WFI_IGNORED

WFI ignored. Fields:

DISABLED bool
: This WFI was ignored because WFI is disabled.

INST_COUNT unsigned int
: Ticks count when ignoring WFI.

REASON enum
: specifies reason why WFI trace was ignored

TRAPPED bool
: This WFI was trapped.

WFI_START

WFI entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFI.

WFI_WAKEUP

WFI wakeup. Fields:

INST_COUNT unsigned int
: Ticks count when WFI wakeup occurred.

REASON enum
: Reason for wakeup.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.21 ARM_Cortex-A76 - trace

This section describes the trace sources.

ASYNC_MEMORY_FAULT

Context ID Register write. Fields:

FAULT unsigned int
: Fault status in ESR format

PADDR unsigned int
: Physical Address (or 0 if unavailable)

VADDR signed int
: Virtual Address (or 0 if unavailable)

ATOMIC_END_ACCESS

Bus trace access for atomics. Fields:

ACCESS_FAIL bool
: Memory access failed

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

LOAD_VALUE unsigned int
: Loaded values

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ATOMIC_START_ACCESS

Bus trace access for atomics. Fields:

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

NSDESC unsigned int
: The security state of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ArchMsg.Error.exit_code

Fields:

EXITCODE signed int
: Exit code to be returned

KIND string
: Component kind that invoked the exit code trace

REASON string
: Reason why the exit code trace was invoked (optional)

ArchMsg.Warning.IgnoredWarmResetRequest

Event fires when software attempts to request warm reset through write to deprecated DBGPRCR.CWRR.

ArchMsg.Warning.RAS_ErrselrUnpredictableState

ERRSELR is in unpredictable state due to value greater than number_of_error_records written to it.

ArchMsg.Warning.RAS_UnpredictableStateWrite

Fields:

REG_NAME string
: Name of register being written.

ArchMsg.Warning.TarmacReconstructor

Unexpected event happened during Tarmac reconstruction. Fields:

REASON string
: Reason

ArchMsg.Warning.Unpredictably Indexed PM Event Register

Fields:

IsDirect unsigned int
: Direct Access rather than indirect

N unsigned int
: Number of Accessible Registers

SEL unsigned int
: Selector

ArchMsg.Warning.branch_to_unaligned_address

Fields:

ADDR unsigned int
: Unaligned branch address

ArchMsg.Warning.cache_contents_unknown

A cache has been activated without being invalidated and may contain UNKNOWN content. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

REGIME unsigned int
: Current translation regime

SIDE unsigned int
: 0 d-side cache or unified cache, 1 i-side

ArchMsg.Warning.dcimvac_matches_watchpoint

Watchpoints matching an AArch32 DCIMVA are implementation defined.

ArchMsg.Warning.decode_fieldmismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_fracbitsnegative

VCVT Instruction is unpredictable with a negative number of fraction bits.

ArchMsg.Warning.decode_initblock

Instruction is unpredictable in an IT block.

ArchMsg.Warning.decode_initblocknotlast

Instruction is unpredictable in an IT block when not the last.

ArchMsg.Warning.decode_invalid_condition

Instruction is unpredictable when the condition is not AL.

ArchMsg.Warning.decode_invalidfieldcombination

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_invalidvalue

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_mem_hint_unallocated

There is no memory hint allocated to this bit pattern.

ArchMsg.Warning.decode_registermatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_registermismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registeroutofrange

Float point instruction is unpredictable indexing beyond the end of the register bank.

ArchMsg.Warning.decode_sbzsbo

Reserved bits in the instruction are not canonical.

ArchMsg.Warning.decode_transactiontoobig

Attempt to perform a memory transaction of more than 128 bytes.

ArchMsg.Warning.decode_unpred_other

Instruction is unpredictable.

ArchMsg.Warning.decode_unpreduseofpc

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13inreglist

Instruction is unpredictable with R13 in register-list.

ArchMsg.Warning.decode_useofd16orhigher

Attempt to use register D16 or higher where it is not allowed.

ArchMsg.Warning.decode_writebackandbaseinlist

Instruction is unpredictable with write back when the base register is in the transfer list.

ArchMsg.Warning.decode_zeroregistersinlist

Instruction is unpredictable when no registers are to be transfered.

ArchMsg.Warning.recursive_branch

An instruction is performing a branch that targets the same instruction. If this was intended then a WFI might be more effective. Fields:

PC unsigned int
: Address of instruction causing the branch

ArchMsg.Warning.recursive_exception

An instruction has generated an exception that targets the same instruction. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CPSR unsigned int
: Processor state

PC unsigned int
: Address of instruction causing the exception

TYPE enum
: Type of exception

ArchMsg.Warning.reentrant_vector_catch

Fields:

ADDR unsigned int
: vector address

ArchMsg.Warning.reserved_it_state

Fields:

ITSTATE unsigned int
: reserved IT state

ArchMsg.Warning.tlb_contents_unknown

Translation has been enabled without invalidating the TLB. Fields:

EL unsigned int
: Exception level of translation regime

INVALIDITY unsigned int
: Which TLB may be invalid (if TLBs are separate)

NON_SECURE unsigned int
: Security state of translation regime

VMID unsigned int
: Current VMID, if applicable

ArchMsg.Warning.unknown_DLR_DSPSR

Exiting debug state with unknown DLR or DSPSR.

ArchMsg.Warning.unknown_ELR_SPSR

Returning from debug exception with unknown ELR or SPSR.

ArchMsg.Warning.unpredictable_a32_breakpoint

Fields:

ADDR1 unsigned int
: Address

ADDR2 unsigned int
: Breakpoint address

BAS unsigned int
: BAS Field

IS_ADDRESS_MISMATCH_BREAKPOINT bool
: Is Mismatch breakpoint, not a match breakpoint

N unsigned int
: Breakpoint number

OFFSET unsigned int
: Indicates which halfword actually considering into the A32 instruction

ArchMsg.Warning.unpredictable_banked_register_access

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

R unsigned int
: R bit

SYSM unsigned int
: SYSm

ArchMsg.Warning.unpredictable_ccfail_undef

Fields:

OPCODE unsigned int
: opcode of the instruction

VADDR unsigned int
: Virtual address of the instruction being executed

ArchMsg.Warning.unpredictable_ici_bits

ICI bits do not apply or are architecturally unpredictable for instruction.

ArchMsg.Warning.unpredictable_t32_breakpoint

Fields:

ADDR1 unsigned int
: Address

ADDR2 unsigned int
: Breakpoint address

BAS unsigned int
: BAS Field

IS_ADDRESS_MISMATCH_BREAKPOINT bool
: Is Mismatch breakpoint, not a match breakpoint

N unsigned int
: Breakpoint number

OFFSET unsigned int
: Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_t32_vector_catch

Fields:

ADDR unsigned int
: Address

OFFSET unsigned int
: Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_watchpoint_far

Fields:

LOWER_BOUND unsigned int
: the lowest address accessed by the instruction that triggered the watchpoint

UPPER_BOUND unsigned int
: the highest watchpointed address accessed by that instruction

ArchMsg.Warning.warning_AdvSIMDExpandImmUnexpectedZero

AdvSIMDExpandImm may treat this immediate value as UNPREDICTABLE.

ArchMsg.Warning.warning_ConditionalSMC

SMC instruction has UNPREDICTABLE effects when conditional when combined with traps.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_access_crosses_page_boundary

Access crosses page boundary.

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault_on_both_pages

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_spanning_different_memory

Fields:

ADDR unsigned int
: address of access crossing page

MEMTYPE_PAGE1 signed int
: First Page's memory type (0-Normal, 1-Non-Normal)

MEMTYPE_PAGE2 signed int
: Second Page's memory type (0-Normal, 1-Non-Normal)

ArchMsg.Warning.warning_access_wraps_around_memory

Fields:

WIDTH unsigned int
: Address width

ArchMsg.Warning.warning_bx_from_thumbee

Fields:

ADDR unsigned int
: Destination address

ArchMsg.Warning.warning_ccsidr_unimplemented_level

Fields:

CSSELR unsigned int
: current effective value of CSSELR

ArchMsg.Warning.warning_change_to_ns_when_tge_set

Attempting to change NS to 1 when HCR.TGE = 1.

ArchMsg.Warning.warning_contiguous_bit_check_abort

Fields:

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ArchMsg.Warning.warning_contiguous_bit_error

Fields:

CONTIG_ADDR unsigned int
: address of the TLB first read, with which this entry is expected to be contiguous

CONTIG_DATA unsigned int
: expected contents based on the entry at CONTIG_ADDR

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ENTRY_DATA unsigned int
: contents of conflicting TLB entry

ArchMsg.Warning.warning_cp10_cp11_mismatch

Fields:

EL unsigned int
: EL owning the control

ArchMsg.Warning.warning_cp10_cp11_reserved_value

Fields:

CP10 unsigned int
: CP10 access

CP11 unsigned int
: CP11 access

ArchMsg.Warning.warning_csselr_level_out_of_range

Fields:

IMPLEMENTED unsigned int
: number of cache levels implemented

LEVEL unsigned int
: written value of CSSELR.Level

TYPE enum
: The type of cache selected by CSSELR.InD and TnD

ArchMsg.Warning.warning_debug_flow_control_bits_not_obeyed

Fields:

ERROR enum
: type of error resulting

REGISTER enum
: register under consideration

ArchMsg.Warning.warning_debug_register_access_during_reset

Fields:

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

ArchMsg.Warning.warning_decode_cps_inconsistent_fields

The mode setting fields in the CPS instruction are inconsistent.

ArchMsg.Warning.warning_decode_invalid_state

Instruction is unpredictable in current exception-level/security state.

ArchMsg.Warning.warning_default_cacheable_mmu_on

Default cacheable is enabled (HCR.DC = 1) while MMU is enabled.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_default_cacheable_vmmu_off

Default cacheable is enabled (HCR.DC = 1) while HCR.VM = 0.

ArchMsg.Warning.warning_deprecated_wvr_bit_2_set

Fields:

N unsigned int
: index

WVR unsigned int
: DBGWVR

ArchMsg.Warning.warning_eret_while_software_step_active_pending

Missing ISB between setting MDSCR_EL1.SS and ERET.

ArchMsg.Warning.warning_exclusive_to_non_normal

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_exclusive_to_non_writeback

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_execute_from_device_memory

An attempt was made to execute from Device memory.

ArchMsg.Warning.warning_illegal_cpsr_mode

Fields:

NEW_MODE unsigned int
: New value of CPSR.M

ArchMsg.Warning.warning_illegal_srs_mode

Fields:

MODE unsigned int
: mode for Banked SP

ArchMsg.Warning.warning_implementation_defined_read_debug_register_in_SCS

Read Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_implementation_defined_sequential_security_transitions_supported

The behaviour of a sequential instruction fetches that cross from non-secure to secure memory and contain SG instruction is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_implementation_defined_stack_limit_check_supported

It is IMPLEMENTATION DEFINED whether stack pointer limit checking is performed for the this instructions. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDRESS unsigned int
: Address of instruction

OPCODE unsigned int
: instruction opcode

ArchMsg.Warning.warning_implementation_defined_write_debug_register_in_SCS

Write Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

DATA unsigned int
: data attempted to be written

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_invalid_tcr_granule

Fields:

REQUEST signed int
: page size requested (or 0 for reserved)

SUBSTITUTE signed int
: best guess available page size

TG_ID bool
: bits TG0 or TG1

ArchMsg.Warning.warning_load_multiple_user_registers_from_user_mode

An LDM(user registers) instruction executed from user mode.

ArchMsg.Warning.warning_load_pc_from_unaligned

PC loaded from an unaligned location.

ArchMsg.Warning.warning_mte_in_mem_attr_not_supported

Fields:

ATTR unsigned int
: Memory attribute

ArchMsg.Warning.warning_reserved_breakpoint_state_match

Fields:

IS_BREAKPOINT unsigned int
: Is breakpoint, not watchpoint

ArchMsg.Warning.warning_shareability

Unpredictable: combination of the 1st and 2nd stages of translation is Normal Inner Non-Cacheable, Outer Non-Cacheable.

ArchMsg.Warning.warning_software_step_set_while_enabled

MDSCR_EL1.SS set to 1 while software step debug exceptions are enabled.

ArchMsg.Warning.warning_thumb_instruction_wraps_around_memory

Thumb instruction warps around memory.

ArchMsg.Warning.warning_ttbr_sbz_bits_are_not_zero

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MSB unsigned int
x-1
TTBR enum
which TTBR register

ArchMsg.Warning.warning_unaligned_address_dbgdtrrx_write

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_address_dbgdtrtx_read

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_device

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_strongly_ordered

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unknown_sau_rnr

SAU_RNR was set to an unsupported value.

ArchMsg.Warning.warning_unpredictable_AIRCR_PRIP_and_BFHFNMINP

The effect of setting both AIRCR.BFHFNMINP and AIRCR.PRIP to 1 is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTCLRACTIVE_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTCLRACTIVE if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_and_SYSRESETREQ

When the processor is halted in Debug state, if a write to the register writes a 1 to both VECTRESET and SYSRESETREQ, the behavior is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTRESET if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_incorrect_VKEY

The value 0x05FA must be written to AIRCR.VKEY, otherwise the register write is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_EXE_RETURN_Reserved_Bit

EXC_RETURN[23:7] are reserved with the special condition that all bits should be written as one. Values other than all 1s are UNPREDICTABLE. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

EXC_RETURN unsigned int
: Special PC value

SBO_MASK unsigned int
: Which bits Should Be One to not UNPRED

ArchMsg.Warning.warning_unpredictable_change_to_DEMCR_MON_STEP_at_insufficient_priority

The effect of changing DEMCR.MON_STEP at an execution priority that is lower than the priority of the DebugMonitor exception is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_change_to_priority_of_active_exception

Changing the priority of an active exception is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_clear_lspact

Fields:

DRVEXC_TAKEN unsigned int
: derived exception taken not pended

REACHED_LAST_STORE unsigned int
: only the last faulted

ArchMsg.Warning.warning_unpredictable_dbg_exit_unaligned_dlr

Debug exit to AArch32 with DLR[0] set to 1. PC[0] can be set to 0 or to DLR[0].

ArchMsg.Warning.warning_unpredictable_exception_catch

Generation of an exception catch event is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_inconsistent_state

state on exception return is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_instruction

exception return instruction is unpredictable.

ArchMsg.Warning.warning_unpredictable_in_debug_state

Instruction is UNPREDICTABLE when executed in debug state.

ArchMsg.Warning.warning_unpredictable_mair_encoding

Fields:

ATTR unsigned int
: Memory attribute

ArchMsg.Warning.warning_unpredictable_pmu_counter_access

Access to PMU counters from non-secure EL0 or EL1 is UNPREDICTABLE with MDCR_EL2.HPMN set to 0.

ArchMsg.Warning.warning_unpredictable_prioritization_breakpoint_match_vector_catch

Two events with the same priority occurred at the same instruction: Address Matching Vector Catch debug event, and Breakpoint debug event It is constrained unpredictable which is taken.

ArchMsg.Warning.warning_unpredictable_stack_selection

UNPREDICTABLE state on context switch.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_tsize_out_of_range

Fields:

TCR enum
: Which TCR

TSIZE signed int
: Value of TSize

TSIZE_MAX signed int
: The maximum allowed value

TTBR unsigned int
: Which TTBR region

ArchMsg.Warning.warning_unpredictable_unaligned_pc_as_base_register

The PC value must be word-aligned, otherwise the behavior of the instruction is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_unaligned_pop_stack

Fields:

FRAME_ADDR unsigned int
: unaligned stack frame address

ArchMsg.Warning.warning_unpredictable_vmsa_memattrib

Fields:

TEXCB unsigned int
: texcb value

TEX_REMAP bool
: use tex remap

ArchMsg.Warning.warning_unpredictable_vtcr_t0sz_sl0

Fields:

SL0 signed int
: VTCR.SL0, starting level for stage 2 translation table walks

T0SZ signed int
: VTCR.T0SZ, required input address range

ArchMsg.Warning.warning_unpredictable_write_DHCSR

Fields:

NEW unsigned int
: data attempted to be written

OLD unsigned int
: previous value

ArchMsg.Warning.warning_unpredictable_write_SHCSR

UNPREDICTABLE setting a pending bit to 1 for an exception with priority >= execution priority.

ArchMsg.Warning.warning_unsupported_access_to_memory_mapped_register

Access to memory mapped register is unsupported.

ArchMsg.Warning.warning_user_jmcr_access

Fields:

WRITE unsigned int
: access is write

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_wcr_mask_and_bas

DBGWCR_EL1.MASK is non-zero and BAS is not 0xff.

ArchMsg.Warning.warning_wcr_mask_reserved

DBGWCR_ELn.MASK is set to a reserved value.

ArchMsg.Warning.warning_wcr_non_configuous_bas

DBGWCR_EL1.BAS has non-contiguous set of ones.

ArchMsg.Warning.warning_wvr_masked_bit_not_zero

DBGWVRn_EL1.VA contains a bit masked by DBGWCRn_EL1.MASK that is not zero.

ArchMsg.Why.why_branch_target_exception

A branch target exception has occurred. Fields:

BTYPE unsigned int
: pstate.BTYPE at time branch taken

OPCODE unsigned int
: opcode at branch target

PC unsigned int
: Virtual Address of branch target

ArchMsg.Why.why_illegal_state

An illegal mode change has occured. Fields:

MESSAGE string
: The message

BRANCH_MISPREDICT

Simulating branch mispredict. Fields:

PC unsigned int
: Origin address (or 0 if unavailable)

BRA_DIR

Direct branches, to immediate address. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

BRA_INDIR

Indirect branches, perhaps to a register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR_CCFAIL

Indirect branches fail, perhaps to a register.

BRA_RET_CCFAIL

Branche return fail, perhaps to a register.

CACHE_MAINTENANCE_OP

Cache Maintenance Operation. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Specified MVA or Set/way

FUNCTION enum
: Clean / Invalidate

SCOPE enum
: Affected region

SIDE enum
: Inst / Data.

CCFAIL

Conditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCFAIL_UNC

Unconditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS

Conditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS_UNC

Unconditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CHECKPOINT_MESSAGE

Report error messages from the checkpointing process. Fields:

message string
: Message contents.

CHECKPOINT_RESTORE_END

Checkpoint restore completed.

CHECKPOINT_RESTORE_START

Checkpoint restore about to start.

CHECKPOINT_SAVE_END

Checkpoint save completed.

CHECKPOINT_SAVE_START

Checkpoint save about to start.

CODE_CACHE_MAINT

Code cache maintenance. Fields:

END_INCL unsigned int
: Inclusive end addres for by VA or PA requests

NS bool
: Non-secure world for PA

START unsigned int
: Start address for by VA or PA requests

TYPE enum
: Request type.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

COMPILE_BLOCK_END

Last instruction of basic block translated. Fields:

VADDR unsigned int
: Address of next instruction after this basic block

COMPILE_BLOCK_START

First instruction of basic block translated. Fields:

VADDR unsigned int
: Address of first instruction in basic block

COMPILE_INST

ARM instruction compiled. Fields:

DISASS string
: Disassembly of the instruction.

ISET enum
: The instruction set of this instruction.

ITSTATE unsigned int
: The ITSTATE current for the instruction.

OPCODE unsigned int
: The opcode of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

CONTEXTIDR

Context ID Register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

NS bool
: Secure or nonsecure banked register is accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

CONTEXT_SYNC

Called for every context synchronization event. Fields:

ARCH_CSE bool
: Architectural Context Synchronization Event

SYSREG_SYNC bool
: System register synchronization event

CORE_ENDIAN

Core BE8 Big-Endian state changed. Fields:

BE8 bool
: Core BE8 Big-Endian state

CORE_INFO

Static processor attributes. Only triggered by a call to DumpState(). Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ARCH_PROFILE enum
: The architecture profile of the core.

CLUSTER_ID unsigned int
: The cluster ID of this processor.

CORE_NUM unsigned int
: The number of this core in an MP processor.

DETAILED_CORE_FEATURES enum
: Detailed core features as a bitfield. Enum values are bitmasks to extract the info from the field

FPU_VERSION enum
: The VFP version implemented by the core.

MEM_ARCH enum
: The memory architecture of the core.

MIDR_EL1 unsigned int
: MIDR_EL1 value

NUM_CORES unsigned int
: The number of cores in this MP processor.

QUANTUM_SIZE unsigned int
: The default quantum size of the core.

SECURITY_FEATURES bool
: Does the core have security features?

CORE_LOADS

Processor load accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CORE_REGS

Changes of the core registers R0 to R14. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number, 0 to 14.

MODE enum
: Bank of the register accessed.

OLD_VALUE unsigned int
: The old value overwritten.

PHYS_ID enum
: The physical register accessed.

VALUE unsigned int
: The new value written to the register.

CORE_REGS64

Changes of the core registers X0..X30, SP_ELn. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: The register number, 0..30 for X0..X30, >=32 for SP_ELn.

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

CORE_STORES

Processor store accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CP14_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP14_WRITE

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CP15_WRITE

System Control Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

CPSR

CPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

OLD_VALUE unsigned int
: The old CPSR value

UNKNOWN unsigned int
: Bits within the register that have unknown value.

VALUE unsigned int
: The new CPSR value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CRYPTO_SPEC

Every crypto instruction speculatively executed.

CorePowerStateChange

Core power-up state has changed. Fields:

PU bool
: Whether the core powered up

DATA_CACHE_SET_ALLOC_TAGS

Set allocation tags from DC GVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of memory in bytes.

TAG unsigned int
: The value of the tag being set.

VADDR unsigned int
: Virtual address used for setting allocation tags.

DATA_CACHE_ZERO

Zero and invalidate cache line from DC ZVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of zero'd memory in bytes.

VADDR unsigned int
: Virtual address of the zero'd cache line.

DEBUG_EVENT

Hardware debug support event. Fields:

EVENT enum
: Description of event

VALUE unsigned int
: data value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

END_COMPILE

Compilation end. Fields:

END_OF_PAGE_COUNT unsigned int
: Number of basic blocks exited due to page boundary since START_COMPILE.

FETCHFAIL_COUNT unsigned int
: Number of basic blocks exited due to ifetch failure START_COMPILE.

INST_COUNT unsigned int
: Number of instructions compiled since START_COMPILE.

NONSEQ_COUNT unsigned int
: Number of basic blocks exited due to non-sequential instructions since START_COMPILE.

PAGE_STRADDLE_COUNT unsigned int
: Number of basic blocks exited due to unaligned instructions crossing page since START_COMPILE.

ESB

Error Synchronization Barrier hit. Fields:

ASYNC bool
: ESB is async

IMPLICIT bool
: ESB is implicit

EXCEPTION

Exception taken. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

IS_PHYSICAL bool
: Physical or Virtual exception

LR unsigned int
: The value assigned to the link register.

PC unsigned int
: The location where the exception occurred.

PREFERRED_RETURN unsigned int
: The preferred return address for the exception.

TARGET_ISET enum
: The instruction set of the exception handler code.

TARGET_PC unsigned int
: The address the exception branches to.

VECTOR enum
: The exception vector.

VECTOR_OFFSET unsigned int
: The offset of an exception vector from the base address.

EXCEPTION_END

Every exception completed.

EXCEPTION_RAISE

Every exception raised.

EXCEPTION_RETURN

Branches on leaving exception. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

EXCEPTION_RETURN_PREBRANCH

Exception return event that must occur before PC and processor state have updated.

EXCEPTION_START

Every exception started.

FIQ_TAKEN

FIQ taken exception.

FP_STATE

Floating point state. Fields:

FPSR unsigned int
: FPSR register value.

MASK unsigned int
: The mask indicating updated cumulative bits.

HLT

HLT instruction occurred. Fields:

IMM16 unsigned int
: The 16-bit immediate value encoded in HLT instruction.

INST

Every instruction executed. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DEBUG_STATE bool
: The instruction is executed in debug state.

DISASS string
: Disassembly of instruction.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The current instruction set.

ITSTATE unsigned int
: The current ITSTATE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

MODE enum
: The mode the core is in.

NS unsigned int
: The core's non-secure bit.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

OPCODE unsigned int
: The opcode of the instruction.

PADDR unsigned int
: The physical address of the instruction.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

INST_END

Every instruction completed.

INST_START

Every instruction started. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

INST_STRADDLE

Instruction straddles boundary. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

IRQ_TAKEN

IRQ taken exception.

ISB

ISB instruction occurred.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOAD_MULTIPLE_REGS_LIST

List of destination registers (not source) of a multiple registers load instruction. Fields:

REG string
: Register to add into the list

LOCAL_MONITOR

Local monitor activity. Fields:

PADDR unsigned int
: Local Monitor Address

State enum
: State of the monitor (Open/Exclusive)

MEMMAP_DEBUG_READ

Memory mapped reads to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

MEMMAP_DEBUG_WRITE

Memory mapped writes to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

MMU_TRANS

Address translation information. Fields:

ASID unsigned int
: Address space identifier.

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Hyp bool
: Entry matches in Hyp state only

INNERCACHE_RA bool
: Is the inner cache allocate on read

INNERCACHE_TRANSIENT bool
: Is the inner write-through transient

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

MEMTYPE enum
: Memory type.

NSDESC enum
: Is secure side supposed to access secure or nonsecure memory

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TRANSIENT bool
: Is the outer write-through transient

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

PADDR unsigned int
: Physical address of the access.

PAGESIZE unsigned int
: Page size as log2(size).

SH enum
: Shareability

SIDE enum
: Inst / Data.

STAGE1_PERM unsigned int
: Stage 1 permissions, [5:3] Privileged access (XWR) [2:0] User access (XWR)

STAGE2_PERM unsigned int
: Stage 2 permission mask [5:3] Privileged access (XWR) [2:0] User access (XWR)

VADDR unsigned int
: Virtual address of the access.

VMID unsigned int
: Virtual machine identifier.

XS bool
: XS Attribute

nG enum
: Flag indicating whether ASID will be matched

MMU_TTB_READ

This event is triggered by reads caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data read.

IPA unsigned int
: For stage 1, the IPA of the read.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

OUTADDR unsigned int
: The output physical address.

PADDR unsigned int
: The physical address of the read.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MMU_TTB_WRITE

This event is triggered by writes caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data written.

IPA unsigned int
: For stage 1, the IPA of the write.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

PADDR unsigned int
: The physical address of the write.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MODE_CHANGE

Mode change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: The new mode.

NON_SECURE enum
: The new security state

OLD_MODE enum
: The old mode.

NOT_TAG_CHECKED_INST

MTE Tag not checked for current instruction.

PERIODIC

Called for every quantum. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

PC unsigned int
: The address of the next instruction to be executed on this CPU.

PMU_COUNTER_OVERFLOW

PMU counter overflow. Fields:

EVENT_ID unsigned int
: Event Identifier

INDEX unsigned int
: Counter index (as selected by PMSELR).

INTERRUPT bool
: Is interrupt enabled on overflow for this counter

PREFETCH_MEMORY64

Prefetch from PRFM or PRFUM instructions. Fields:

PRFOP unsigned int
: Prefetch hint

VADDR unsigned int
: Virtual address of the location that should be prefetched.

PRELOAD_DATA

Data preload from PLD instruction. Fields:

VADDR unsigned int
: Virtual address of the data that should be preloaded.

PRELOAD_INST

Instruction preload from PLI instruction. Fields:

VADDR unsigned int
: Virtual address of the instruction that should be preloaded.

PRE_CORE_LOAD

Trace just before a core load. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_CORE_STORE

Trace just before a core store. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_TTB_READ

This event is triggered before reads caused by a translation table walk. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR unsigned int
The physical address of the read.

PSTATE_PACTIVE_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Operating mode after the transition
OLD_STATE enum
Operating mode before the transition

PSTATE_PACTIVE_OPMODE_TRANSITION_DENIED

Trace the denied pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Operating mode after the transition
OLD_STATE enum
Operating mode before the transition
REASON string
Reason for the transition being denied

PSTATE_PACTIVE_POWER_TRANSITION

Trace pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Power mode after the transition
OLD_STATE enum
Power mode before the transition

PSTATE_PACTIVE_POWER_TRANSITION_DENIED

Trace the denied pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Power mode after the transition
OLD_STATE enum
Power mode before the transition
REASON string
Reason for the transition being denied

PSTATE_PACTIVE_THREAD_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions for threads on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Operating mode after the transition
OLD_STATE enum
Operating mode before the transition

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

RUN_STATE

Run state transition. Fields:

INST_COUNT unsigned int
: Ticks count at point of transition.

NEW enum
: New run state.

OLD enum
: Old run state.

SBP_SAMPLE_COLLISION

Statistical profiling samples collided .

SBP_SAMPLE_DISCARDED

Fields:

BUFFER_POINTER unsigned int
: Value of buffer pointer

DATA unsigned int
: Data discarded

SIZE unsigned int
: Size of sample, in bytes

SBP_SAMPLE_END

Statistical profiling of an operation ended.

SBP_SAMPLE_POP

Operation which might be sampled by statistical profiling.

SBP_SAMPLE_RECORDED

Fields:

BUFFER_ADDRESS unsigned int
: Virtual Address of start of buffer write

DATA unsigned int
: Data written to the buffer

SIZE unsigned int
: Size of sample written to buffer, in bytes

SBP_SAMPLE_START

Statistical profiling of an operation started.

SBP_SAMPLE_TAKEN

Operation sampled by statistical profiling after filtering.

SEMIHOSTING_CALL

Call of a semihost function occurred. Fields:

PC unsigned int
: The program counter after the semihosting call.

SEMIHOSTING_PRECALL

About to call semihost. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

EL2 bool
: Use EL2 translation regime.

NS bool
: Is Non-Secure.

REG_WIDTH unsigned int
: The current register width in bytes.

SIGNAL

External signal state change. Fields:

SIGNAL enum
: Signal that changed

STATE bool
: Signal asserted state

SOFTWARE_STEP

Return the Debug Software Step state. Fields:

STATE enum
: Software Step state

SPSR

SPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: Which of the banked SPSR registers is written.

OLD_VALUE unsigned int
: The old SPSR value

VALUE unsigned int
: The new SPSR value

START_COMPILE

Compilation started. Fields:

VADDR unsigned int
: Instruction where compilation begins.

STG_MULTIPLE_MICRO_OPS_ADDRESS_RANGE

Address range for MTE stores with multiple uops. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

STORE_MULTIPLE_REGS_ADDRESS_RANGE

Address range for stores with multiple registers. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

SUBOPTIMAL_LDST_RETIRED

suboptimal load/store. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1: half-word, 2: word ...)

OPTIMAL_ALIGN_SELECTOR unsigned int
: Selector to decide the optimal alignment size (1: access width, 2: 4-bytes, 3: 8-bytes, ..., 12: 4-kbytes)

VADDR unsigned int
: The virtual address of the access

SYNC

Called for every synchronization, that is every quantum break. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

LOCAL_QUANTUM unsigned int
: The local quantum of this CPU.

LOCAL_TIME unsigned int
: The local time of this CPU.

SYNC_REASON enum
: Reason for sync.

SYSCALL

System call instruction executed. Fields:

IMM unsigned int
: Immediate value of the system call instruction.

TYPE enum
: System call type.

VADDR unsigned int
: Instruction that caused the system call.

SYSREG_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

opc2 unsigned int
: opcode 2

SYSREG_UPDATE32

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE64

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

UNALIGNED_LDST_RETIRED

Processor unaligned load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1=>half-word, 2->word ...)

NUMBER_OF_BEATS unsigned int
: Number of accesses (beats) in this burst (Total size in bytes is ACCESS_SIZE * NUMBER_OF_BEATS)

VADDR unsigned int
: The virtual address of the access

VFP_D_REGS

VFP/NEON D 64 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_Q_REGS

VFP/NEON Q 128 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MASK unsigned int
: Each bit indicates a write to the corresponding byte in the register. For example, 0x1 would indicate a write to VALUE[7:0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_SYS_REGS

Writes to the VFP/NEON units system registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: Which VFP system register is written.

MASK unsigned int
: The mask indicating updated cumulative bits.

OLD_VALUE unsigned int
: The register's old value overwritten.

VALUE unsigned int
: The new value written to the VFP system register.

VFP_S_REGS

VFP/NEON S 32 bit register write. Fields:

ALIASING enum
: Type of register aliasing used.

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

WAYPOINT

Signals end of basic block execution. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: Origin instructions set

IS_COND bool
: Indicates if this is a conditional waypoint.

PC unsigned int
: Origin address (or 0 if unavailable)

TAKEN bool
: Indicates if this waypoint was taken.

TARGET unsigned int
: Destination address

TARGET_ISET enum
: Destination instructions set

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

WFE_END

WFE ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFE.

WFE_EVENT_REGISTER

WFE event register status: set/clear, reason. Fields:

INST_COUNT unsigned int
: Ticks count.

REASON enum
: Reason for set/clear. Only REASON==0 (Cleared by WFE) clears the bit, all other reasons set it.

WFE_IGNORED

WFE ignored. Fields:

EVENT bool
: This WFE was ignored because the event register was set.

INST_COUNT unsigned int
: Ticks count when ignoring WFE.

TRAPPED bool
: This WFE was trapped.

WFE_START

WFE entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFE.

WFI_END

WFI ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFI.

WFI_IGNORED

WFI ignored. Fields:

DISABLED bool
: This WFI was ignored because WFI is disabled.

INST_COUNT unsigned int
: Ticks count when ignoring WFI.

REASON enum
: specifies reason why WFI trace was ignored

TRAPPED bool
: This WFI was trapped.

WFI_START

WFI entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFI.

WFI_WAKEUP

WFI wakeup. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT `unsigned int`
: Ticks count when WFI wakeup occurred.

REASON `enum`
: Reason for wakeup.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.22 ARM_Cortex-A76AE - trace

This section describes the trace sources.

ASYNC_MEMORY_FAULT

Context ID Register write. Fields:

FAULT unsigned int
: Fault status in ESR format

PADDR unsigned int
: Physical Address (or 0 if unavailable)

VADDR signed int
: Virtual Address (or 0 if unavailable)

ATOMIC_END_ACCESS

Bus trace access for atomics. Fields:

ACCESS_FAIL bool
: Memory access failed

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

LOAD_VALUE unsigned int
: Loaded values

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ATOMIC_START_ACCESS

Bus trace access for atomics. Fields:

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

NSDESC unsigned int
: The security state of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ArchMsg.Error.exit_code

Fields:

EXITCODE signed int
: Exit code to be returned

KIND string
: Component kind that invoked the exit code trace

REASON string
: Reason why the exit code trace was invoked (optional)

ArchMsg.Warning.IgnoredWarmResetRequest

Event fires when software attempts to request warm reset through write to deprecated DBGPRCR.CWRR.

ArchMsg.Warning.RAS_ErrselrUnpredictableState

ERRSELR is in unpredictable state due to value greater than number_of_error_records written to it.

ArchMsg.Warning.RAS_UnpredictableStateWrite

Fields:

REG_NAME string
: Name of register being written.

ArchMsg.Warning.TarmacReconstructor

Unexpected event happened during Tarmac reconstruction. Fields:

REASON string
: Reason

ArchMsg.Warning.Unpredictably Indexed PM Event Register

Fields:

IsDirect unsigned int
: Direct Access rather than indirect

N unsigned int
: Number of Accessible Registers

SEL unsigned int
: Selector

ArchMsg.Warning.branch_to_unaligned_address

Fields:

ADDR unsigned int
: Unaligned branch address

ArchMsg.Warning.cache_contents_unknown

A cache has been activated without being invalidated and may contain UNKNOWN content. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

REGIME unsigned int
: Current translation regime

SIDE unsigned int
: 0 d-side cache or unified cache, 1 i-side

ArchMsg.Warning.dcimvac_matches_watchpoint

Watchpoints matching an AArch32 DCIMVA are implementation defined.

ArchMsg.Warning.decode_fieldmismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_fracbitsnegative

VCVT Instruction is unpredictable with a negative number of fraction bits.

ArchMsg.Warning.decode_initblock

Instruction is unpredictable in an IT block.

ArchMsg.Warning.decode_initblocknotlast

Instruction is unpredictable in an IT block when not the last.

ArchMsg.Warning.decode_invalid_condition

Instruction is unpredictable when the condition is not AL.

ArchMsg.Warning.decode_invalidfieldcombination

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_invalidvalue

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_mem_hint_unallocated

There is no memory hint allocated to this bit pattern.

ArchMsg.Warning.decode_registermatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_registermismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registeroutofrange

Float point instruction is unpredictable indexing beyond the end of the register bank.

ArchMsg.Warning.decode_sbzsbo

Reserved bits in the instruction are not canonical.

ArchMsg.Warning.decode_transactiontoobig

Attempt to perform a memory transaction of more than 128 bytes.

ArchMsg.Warning.decode_unpred_other

Instruction is unpredictable.

ArchMsg.Warning.decode_unpreduseofpc

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13inreglist

Instruction is unpredictable with R13 in register-list.

ArchMsg.Warning.decode_useofd16orhigher

Attempt to use register D16 or higher where it is not allowed.

ArchMsg.Warning.decode_writebackandbaseinlist

Instruction is unpredictable with write back when the base register is in the transfer list.

ArchMsg.Warning.decode_zeroregistersinlist

Instruction is unpredictable when no registers are to be transfered.

ArchMsg.Warning.recursive_branch

An instruction is performing a branch that targets the same instruction. If this was intended then a WFI might be more effective. Fields:

PC unsigned int
: Address of instruction causing the branch

ArchMsg.Warning.recursive_exception

An instruction has generated an exception that targets the same instruction. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CPSR unsigned int
: Processor state

PC unsigned int
: Address of instruction causing the exception

TYPE enum
: Type of exception

ArchMsg.Warning.reentrant_vector_catch

Fields:

ADDR unsigned int
: vector address

ArchMsg.Warning.reserved_it_state

Fields:

ITSTATE unsigned int
: reserved IT state

ArchMsg.Warning.tlb_contents_unknown

Translation has been enabled without invalidating the TLB. Fields:

EL unsigned int
: Exception level of translation regime

INVALIDITY unsigned int
: Which TLB may be invalid (if TLBs are separate)

NON_SECURE unsigned int
: Security state of translation regime

VMID unsigned int
: Current VMID, if applicable

ArchMsg.Warning.unknown_DLR_DSPSR

Exiting debug state with unknown DLR or DSPSR.

ArchMsg.Warning.unknown_ELR_SPSR

Returning from debug exception with unknown ELR or SPSR.

ArchMsg.Warning.unpredictable_a32_breakpoint

Fields:

ADDR1 unsigned int
: Address

ADDR2 unsigned int
: Breakpoint address

BAS unsigned int
: BAS Field

IS_ADDRESS_MISMATCH_BREAKPOINT bool
: Is Mismatch breakpoint, not a match breakpoint

N unsigned int
: Breakpoint number

OFFSET unsigned int
: Indicates which halfword actually considering into the A32 instruction

ArchMsg.Warning.unpredictable_banked_register_access

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

R unsigned int
R bit
SYSM unsigned int
SYSm

ArchMsg.Warning.unpredictable_ccfail_undef

Fields:

OPCODE unsigned int
opcode of the instruction
VADDR unsigned int
Virtual address of the instruction being executed

ArchMsg.Warning.unpredictable_ici_bits

ICI bits do not apply or are architecturally unpredictable for instruction.

ArchMsg.Warning.unpredictable_t32_breakpoint

Fields:

ADDR1 unsigned int
Address
ADDR2 unsigned int
Breakpoint address
BAS unsigned int
BAS Field
IS_ADDRESS_MISMATCH_BREAKPOINT bool
Is Mismatch breakpoint, not a match breakpoint
N unsigned int
Breakpoint number
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_t32_vector_catch

Fields:

ADDR unsigned int
Address
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_watchpoint_far

Fields:

LOWER_BOUND unsigned int
the lowest address accessed by the instruction that triggered the watchpoint
UPPER_BOUND unsigned int
the highest watchpointed address accessed by that instruction

ArchMsg.Warning.warning_AdvSIMDExpandImmUnexpectedZero

AdvSIMDExpandImm may treat this immediate value as UNPREDICTABLE.

ArchMsg.Warning.warning_ConditionalSMC

SMC instruction has UNPREDICTABLE effects when conditional when combined with traps.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_access_crosses_page_boundary

Access crosses page boundary.

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault_on_both_pages

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_spanning_different_memory

Fields:

ADDR `unsigned int`
: address of access crossing page

MEMTYPE_PAGE1 `signed int`
: First Page's memory type (0-Normal, 1-Non-Normal)

MEMTYPE_PAGE2 `signed int`
: Second Page's memory type (0-Normal, 1-Non-Normal)

ArchMsg.Warning.warning_access_wraps_around_memory

Fields:

WIDTH `unsigned int`
: Address width

ArchMsg.Warning.warning_bx_from_thumbee

Fields:

ADDR `unsigned int`
: Destination address

ArchMsg.Warning.warning_ccsidr_unimplemented_level

Fields:

CSSELR `unsigned int`
: current effective value of CSSELR

ArchMsg.Warning.warning_change_to_ns_when_tge_set

Attempting to change NS to 1 when HCR.TGE = 1.

ArchMsg.Warning.warning_contiguous_bit_check_abort

Fields:

ENTRY_ADDR `unsigned int`
: address of conflicting TLB entry

ArchMsg.Warning.warning_contiguous_bit_error

Fields:

CONTIG_ADDR `unsigned int`
: address of the TLB first read, with which this entry is expected to be contiguous

CONTIG_DATA `unsigned int`
: expected contents based on the entry at CONTIG_ADDR

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ENTRY_DATA unsigned int
: contents of conflicting TLB entry

ArchMsg.Warning.warning_cp10_cp11_mismatch

Fields:

EL unsigned int
: EL owning the control

ArchMsg.Warning.warning_cp10_cp11_reserved_value

Fields:

CP10 unsigned int
: CP10 access

CP11 unsigned int
: CP11 access

ArchMsg.Warning.warning_csselr_level_out_of_range

Fields:

IMPLEMENTED unsigned int
: number of cache levels implemented

LEVEL unsigned int
: written value of CSSELR.Level

TYPE enum
: The type of cache selected by CSSELR.InD and TnD

ArchMsg.Warning.warning_debug_flow_control_bits_not_obeyed

Fields:

ERROR enum
: type of error resulting

REGISTER enum
: register under consideration

ArchMsg.Warning.warning_debug_register_access_during_reset

Fields:

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

ArchMsg.Warning.warning_decode_cps_inconsistent_fields

The mode setting fields in the CPS instruction are inconsistent.

ArchMsg.Warning.warning_decode_invalid_state

Instruction is unpredictable in current exception-level/security state.

ArchMsg.Warning.warning_default_cacheable_mmu_on

Default cacheable is enabled (HCR.DC = 1) while MMU is enabled.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_default_cacheable_vmmu_off

Default cacheable is enabled (HCR.DC = 1) while HCR.VM = 0.

ArchMsg.Warning.warning_deprecated_wvr_bit_2_set

Fields:

N unsigned int
: index

WVR unsigned int
: DBGWVR

ArchMsg.Warning.warning_eret_while_software_step_active_pending

Missing ISB between setting MDSCR_EL1.SS and ERET.

ArchMsg.Warning.warning_exclusive_to_non_normal

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_exclusive_to_non_writeback

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_execute_from_device_memory

An attempt was made to execute from Device memory.

ArchMsg.Warning.warning_illegal_cpsr_mode

Fields:

NEW_MODE unsigned int
: New value of CPSR.M

ArchMsg.Warning.warning_illegal_srs_mode

Fields:

MODE unsigned int
: mode for Banked SP

ArchMsg.Warning.warning_implementation_defined_read_debug_register_in_SCS

Read Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_implementation_defined_sequential_security_transitions_supported

The behaviour of a sequential instruction fetches that cross from non-secure to secure memory and contain SG instruction is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_implementation_defined_stack_limit_check_supported

It is IMPLEMENTATION DEFINED whether stack pointer limit checking is performed for the this instructions. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDRESS unsigned int
: Address of instruction

OPCODE unsigned int
: instruction opcode

ArchMsg.Warning.warning_implementation_defined_write_debug_register_in_SCS

Write Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

DATA unsigned int
: data attempted to be written

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_invalid_tcr_granule

Fields:

REQUEST signed int
: page size requested (or 0 for reserved)

SUBSTITUTE signed int
: best guess available page size

TG_ID bool
: bits TG0 or TG1

ArchMsg.Warning.warning_load_multiple_user_registers_from_user_mode

An LDM(user registers) instruction executed from user mode.

ArchMsg.Warning.warning_load_pc_from_unaligned

PC loaded from an unaligned location.

ArchMsg.Warning.warning_mte_in_mem_attr_not_supported

Fields:

ATTR unsigned int
: Memory attribute

ArchMsg.Warning.warning_reserved_breakpoint_state_match

Fields:

IS_BREAKPOINT unsigned int
: Is breakpoint, not watchpoint

ArchMsg.Warning.warning_shareability

Unpredictable: combination of the 1st and 2nd stages of translation is Normal Inner Non-Cacheable, Outer Non-Cacheable.

ArchMsg.Warning.warning_software_step_set_while_enabled

MDSCR_EL1.SS set to 1 while software step debug exceptions are enabled.

ArchMsg.Warning.warning_thumb_instruction_wraps_around_memory

Thumb instruction warps around memory.

ArchMsg.Warning.warning_ttbr_sbz_bits_are_not_zero

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MSB unsigned int
x-1
TTBR enum
which TTBR register

ArchMsg.Warning.warning_unaligned_address_dbgdtrrx_write

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_address_dbgdtrtx_read

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_device

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_strongly_ordered

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unknown_sau_rnr

SAU_RNR was set to an unsupported value.

ArchMsg.Warning.warning_unpredictable_AIRCR_PRIP_and_BFHFNMINP

The effect of setting both AIRCR.BFHFNMINP and AIRCR.PRIP to 1 is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTCLRACTIVE_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTCLRACTIVE if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_and_SYSRESETREQ

When the processor is halted in Debug state, if a write to the register writes a 1 to both VECTRESET and SYSRESETREQ, the behavior is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTRESET if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_incorrect_VKEY

The value 0x05FA must be written to AIRCR.VKEY, otherwise the register write is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_EXE_RETURN_Reserved_Bit

EXC_RETURN[23:7] are reserved with the special condition that all bits should be written as one. Values other than all 1s are UNPREDICTABLE. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

EXC_RETURN unsigned int
Special PC value
SBO_MASK unsigned int
Which bits Should Be One to not UNPRED

ArchMsg.Warning.warning_unpredictable_change_to_DEMCR_MON_STEP_at_insufficient_priority

The effect of changing DEMCR.MON_STEP at an execution priority that is lower than the priority of the DebugMonitor exception is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_change_to_priority_of_active_exception

Changing the priority of an active exception is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_clear_lspact

Fields:

DRVEXC_TAKEN unsigned int
derived exception taken not pended
REACHED_LAST_STORE unsigned int
only the last faulted

ArchMsg.Warning.warning_unpredictable_dbg_exit_unaligned_dlr

Debug exit to AArch32 with DLR[0] set to 1. PC[0] can be set to 0 or to DLR[0].

ArchMsg.Warning.warning_unpredictable_exception_catch

Generation of an exception catch event is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_inconsistent_state

state on exception return is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_instruction

exception return instruction is unpredictable.

ArchMsg.Warning.warning_unpredictable_in_debug_state

Instruction is UNPREDICTABLE when executed in debug state.

ArchMsg.Warning.warning_unpredictable_mair_encoding

Fields:

ATTR unsigned int
Memory attribute

ArchMsg.Warning.warning_unpredictable_pmu_counter_access

Access to PMU counters from non-secure EL0 or EL1 is UNPREDICTABLE with MDCR_EL2.HPMN set to 0.

ArchMsg.Warning.warning_unpredictable_prioritization_breakpoint_match_vector_catch

Two events with the same priority occurred at the same instruction: Address Matching Vector Catch debug event, and Breakpoint debug event It is constrained unpredictable which is taken.

ArchMsg.Warning.warning_unpredictable_stack_selection

UNPREDICTABLE state on context switch.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_tsize_out_of_range

Fields:

TCR enum
: Which TCR

TSIZE signed int
: Value of TSize

TSIZE_MAX signed int
: The maximum allowed value

TTBR unsigned int
: Which TTBR region

ArchMsg.Warning.warning_unpredictable_unaligned_pc_as_base_register

The PC value must be word-aligned, otherwise the behavior of the instruction is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_unaligned_pop_stack

Fields:

FRAME_ADDR unsigned int
: unaligned stack frame address

ArchMsg.Warning.warning_unpredictable_vmsa_memattrib

Fields:

TEXCB unsigned int
: texcb value

TEX_REMAP bool
: use tex remap

ArchMsg.Warning.warning_unpredictable_vtcr_t0sz_sl0

Fields:

SL0 signed int
: VTCR.SL0, starting level for stage 2 translation table walks

T0SZ signed int
: VTCR.T0SZ, required input address range

ArchMsg.Warning.warning_unpredictable_write_DHCSR

Fields:

NEW unsigned int
: data attempted to be written

OLD unsigned int
: previous value

ArchMsg.Warning.warning_unpredictable_write_SHCSR

UNPREDICTABLE setting a pending bit to 1 for an exception with priority >= execution priority.

ArchMsg.Warning.warning_unsupported_access_to_memory_mapped_register

Access to memory mapped register is unsupported.

ArchMsg.Warning.warning_user_jmcr_access

Fields:

WRITE unsigned int
: access is write

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_wcr_mask_and_bas

DBGWCR_EL1.MASK is non-zero and BAS is not 0xff.

ArchMsg.Warning.warning_wcr_mask_reserved

DBGWCR_ELn.MASK is set to a reserved value.

ArchMsg.Warning.warning_wcr_non_configuous_bas

DBGWCR_EL1.BAS has non-contiguous set of ones.

ArchMsg.Warning.warning_wvr_masked_bit_not_zero

DBGWVRn_EL1.VA contains a bit masked by DBGWCRn_EL1.MASK that is not zero.

ArchMsg.Why.why_branch_target_exception

A branch target exception has occurred. Fields:

BTYPE unsigned int
: pstate.BTYPE at time branch taken

OPCODE unsigned int
: opcode at branch target

PC unsigned int
: Virtual Address of branch target

ArchMsg.Why.why_illegal_state

An illegal mode change has occured. Fields:

MESSAGE string
: The message

BRANCH_MISPREDICT

Simulating branch mispredict. Fields:

PC unsigned int
: Origin address (or 0 if unavailable)

BRA_DIR

Direct branches, to immediate address. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

BRA_INDIR

Indirect branches, perhaps to a register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR_CCFAIL

Indirect branches fail, perhaps to a register.

BRA_RET_CCFAIL

Branche return fail, perhaps to a register.

CACHE_MAINTENANCE_OP

Cache Maintenance Operation. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Specified MVA or Set/way

FUNCTION enum
: Clean / Invalidate

SCOPE enum
: Affected region

SIDE enum
: Inst / Data.

CCFAIL

Conditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCFAIL_UNC

Unconditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS

Conditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS_UNC

Unconditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CHECKPOINT_MESSAGE

Report error messages from the checkpointing process. Fields:

message string
: Message contents.

CHECKPOINT_RESTORE_END

Checkpoint restore completed.

CHECKPOINT_RESTORE_START

Checkpoint restore about to start.

CHECKPOINT_SAVE_END

Checkpoint save completed.

CHECKPOINT_SAVE_START

Checkpoint save about to start.

CODE_CACHE_MAINT

Code cache maintenance. Fields:

END_INCL unsigned int
: Inclusive end addres for by VA or PA requests

NS bool
: Non-secure world for PA

START unsigned int
: Start address for by VA or PA requests

TYPE enum
: Request type.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

COMPILE_BLOCK_END

Last instruction of basic block translated. Fields:

VADDR unsigned int
: Address of next instruction after this basic block

COMPILE_BLOCK_START

First instruction of basic block translated. Fields:

VADDR unsigned int
: Address of first instruction in basic block

COMPILE_INST

ARM instruction compiled. Fields:

DISASS string
: Disassembly of the instruction.

ISET enum
: The instruction set of this instruction.

ITSTATE unsigned int
: The ITSTATE current for the instruction.

OPCODE unsigned int
: The opcode of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

CONTEXTIDR

Context ID Register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

NS bool
: Secure or nonsecure banked register is accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

CONTEXT_SYNC

Called for every context synchronization event. Fields:

ARCH_CSE bool
: Architectural Context Synchronization Event

SYSREG_SYNC bool
: System register synchronization event

CORE_ENDIAN

Core BE8 Big-Endian state changed. Fields:

BE8 bool
: Core BE8 Big-Endian state

CORE_INFO

Static processor attributes. Only triggered by a call to DumpState(). Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ARCH_PROFILE enum
: The architecture profile of the core.

CLUSTER_ID unsigned int
: The cluster ID of this processor.

CORE_NUM unsigned int
: The number of this core in an MP processor.

DETAILED_CORE_FEATURES enum
: Detailed core features as a bitfield. Enum values are bitmasks to extract the info from the field

FPU_VERSION enum
: The VFP version implemented by the core.

MEM_ARCH enum
: The memory architecture of the core.

MIDR_EL1 unsigned int
: MIDR_EL1 value

NUM_CORES unsigned int
: The number of cores in this MP processor.

QUANTUM_SIZE unsigned int
: The default quantum size of the core.

SECURITY_FEATURES bool
: Does the core have security features?

CORE_LOADS

Processor load accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CORE_REGS

Changes of the core registers R0 to R14. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number, 0 to 14.

MODE enum
: Bank of the register accessed.

OLD_VALUE unsigned int
: The old value overwritten.

PHYS_ID enum
: The physical register accessed.

VALUE unsigned int
: The new value written to the register.

CORE_REGS64

Changes of the core registers X0..X30, SP_ELn. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: The register number, 0..30 for X0..X30, >=32 for SP_ELn.

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

CORE_STORES

Processor store accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CP14_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP14_WRITE

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CP15_WRITE

System Control Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

CPSR

CPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

OLD_VALUE unsigned int
: The old CPSR value

UNKNOWN unsigned int
: Bits within the register that have unknown value.

VALUE unsigned int
: The new CPSR value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CRYPTO_SPEC

Every crypto instruction speculatively executed.

CorePowerStateChange

Core power-up state has changed. Fields:

PU `bool`
: Whether the core powered up

DATA_CACHE_SET_ALLOC_TAGS

Set allocation tags from DC GVA instruction. Fields:

ATTR `unsigned int`
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC `unsigned int`
: The security state of the access.

PADDR `unsigned int`
: The physical address of the access.

RESPONSE `enum`
: 0=Aborted, 1=OK

SIZE `unsigned int`
: The size of memory in bytes.

TAG `unsigned int`
: The value of the tag being set.

VADDR `unsigned int`
: Virtual address used for setting allocation tags.

DATA_CACHE_ZERO

Zero and invalidate cache line from DC ZVA instruction. Fields:

ATTR `unsigned int`
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC `unsigned int`
: The security state of the access.

PADDR `unsigned int`
: The physical address of the access.

RESPONSE `enum`
: 0=Aborted, 1=OK

SIZE `unsigned int`
: The size of zero'd memory in bytes.

VADDR `unsigned int`
: Virtual address of the zero'd cache line.

DEBUG_EVENT

Hardware debug support event. Fields:

EVENT `enum`
: Description of event

VALUE `unsigned int`
: data value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

END_COMPILE

Compilation end. Fields:

END_OF_PAGE_COUNT unsigned int
: Number of basic blocks exited due to page boundary since START_COMPILE.

FETCHFAIL_COUNT unsigned int
: Number of basic blocks exited due to ifetch failure START_COMPILE.

INST_COUNT unsigned int
: Number of instructions compiled since START_COMPILE.

NONSEQ_COUNT unsigned int
: Number of basic blocks exited due to non-sequential instructions since START_COMPILE.

PAGE_STRADDLE_COUNT unsigned int
: Number of basic blocks exited due to unaligned instructions crossing page since START_COMPILE.

ESB

Error Synchronization Barrier hit. Fields:

ASYNC bool
: ESB is async

IMPLICIT bool
: ESB is implicit

EXCEPTION

Exception taken. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

IS_PHYSICAL bool
: Physical or Virtual exception

LR unsigned int
: The value assigned to the link register.

PC unsigned int
: The location where the exception occurred.

PREFERRED_RETURN unsigned int
: The preferred return address for the exception.

TARGET_ISET enum
: The instruction set of the exception handler code.

TARGET_PC unsigned int
: The address the exception branches to.

VECTOR enum
: The exception vector.

VECTOR_OFFSET unsigned int
: The offset of an exception vector from the base address.

EXCEPTION_END

Every exception completed.

EXCEPTION_RAISE

Every exception raised.

EXCEPTION_RETURN

Branches on leaving exception. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

EXCEPTION_RETURN_PREBRANCH

Exception return event that must occur before PC and processor state have updated.

EXCEPTION_START

Every exception started.

FIQ_TAKEN

FIQ taken exception.

FP_STATE

Floating point state. Fields:

FPSR unsigned int
: FPSR register value.

MASK unsigned int
: The mask indicating updated cumulative bits.

HLT

HLT instruction occurred. Fields:

IMM16 unsigned int
: The 16-bit immediate value encoded in HLT instruction.

INST

Every instruction executed. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DEBUG_STATE bool
: The instruction is executed in debug state.

DISASS string
: Disassembly of instruction.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The current instruction set.

ITSTATE unsigned int
: The current ITSTATE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

MODE enum
: The mode the core is in.

NS unsigned int
: The core's non-secure bit.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

OPCODE unsigned int
: The opcode of the instruction.

PADDR unsigned int
: The physical address of the instruction.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

INST_END

Every instruction completed.

INST_START

Every instruction started. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

INST_STRADDLE

Instruction straddles boundary. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

IRQ_TAKEN

IRQ taken exception.

ISB

ISB instruction occurred.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOAD_MULTIPLE_REGS_LIST

List of destination registers (not source) of a multiple registers load instruction. Fields:

REG string
: Register to add into the list

LOCAL_MONITOR

Local monitor activity. Fields:

PADDR unsigned int
: Local Monitor Address

State enum
: State of the monitor (Open/Exclusive)

MEMMAP_DEBUG_READ

Memory mapped reads to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

MEMMAP_DEBUG_WRITE

Memory mapped writes to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

MMU_TRANS

Address translation information. Fields:

ASID unsigned int
: Address space identifier.

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Hyp bool
: Entry matches in Hyp state only

INNERCACHE_RA bool
: Is the inner cache allocate on read

INNERCACHE_TRANSIENT bool
: Is the inner write-through transient

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

MEMTYPE enum
: Memory type.

NSDESC enum
: Is secure side supposed to access secure or nonsecure memory

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TRANSIENT bool
: Is the outer write-through transient

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

PADDR unsigned int
: Physical address of the access.

PAGESIZE unsigned int
: Page size as log2(size).

SH enum
: Shareability

SIDE enum
: Inst / Data.

STAGE1_PERM unsigned int
: Stage 1 permissions, [5:3] Privileged access (XWR) [2:0] User access (XWR)

STAGE2_PERM unsigned int
: Stage 2 permission mask [5:3] Privileged access (XWR) [2:0] User access (XWR)

VADDR unsigned int
: Virtual address of the access.

VMID unsigned int
: Virtual machine identifier.

XS bool
: XS Attribute

nG enum
: Flag indicating whether ASID will be matched

MMU_TTB_READ

This event is triggered by reads caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data read.

IPA unsigned int
: For stage 1, the IPA of the read.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

OUTADDR unsigned int
: The output physical address.

PADDR unsigned int
: The physical address of the read.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MMU_TTB_WRITE

This event is triggered by writes caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data written.

IPA unsigned int
: For stage 1, the IPA of the write.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

PADDR unsigned int
: The physical address of the write.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MODE_CHANGE

Mode change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: The new mode.

NON_SECURE enum
: The new security state

OLD_MODE enum
: The old mode.

NOT_TAG_CHECKED_INST

MTE Tag not checked for current instruction.

PERIODIC

Called for every quantum. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

PC unsigned int
: The address of the next instruction to be executed on this CPU.

PMU_COUNTER_OVERFLOW

PMU counter overflow. Fields:

EVENT_ID unsigned int
: Event Identifier

INDEX unsigned int
: Counter index (as selected by PMSELR).

INTERRUPT bool
: Is interrupt enabled on overflow for this counter

PREFETCH_MEMORY64

Prefetch from PRFM or PRFUM instructions. Fields:

PRFOP unsigned int
: Prefetch hint

VADDR unsigned int
: Virtual address of the location that should be prefetched.

PRELOAD_DATA

Data preload from PLD instruction. Fields:

VADDR unsigned int
: Virtual address of the data that should be preloaded.

PRELOAD_INST

Instruction preload from PLI instruction. Fields:

VADDR unsigned int
: Virtual address of the instruction that should be preloaded.

PRE_CORE_LOAD

Trace just before a core load. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_CORE_STORE

Trace just before a core store. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_TTB_READ

This event is triggered before reads caused by a translation table walk. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR unsigned int
The physical address of the read.

PSTATE_PACTIVE_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Operating mode after the transition
OLD_STATE enum
Operating mode before the transition

PSTATE_PACTIVE_OPMODE_TRANSITION_DENIED

Trace the denied pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Operating mode after the transition
OLD_STATE enum
Operating mode before the transition
REASON string
Reason for the transition being denied

PSTATE_PACTIVE_POWER_TRANSITION

Trace pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Power mode after the transition
OLD_STATE enum
Power mode before the transition

PSTATE_PACTIVE_POWER_TRANSITION_DENIED

Trace the denied pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Power mode after the transition
OLD_STATE enum
Power mode before the transition
REASON string
Reason for the transition being denied

PSTATE_PACTIVE_THREAD_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions for threads on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Operating mode after the transition
OLD_STATE enum
Operating mode before the transition

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

RUN_STATE

Run state transition. Fields:

INST_COUNT unsigned int
: Ticks count at point of transition.

NEW enum
: New run state.

OLD enum
: Old run state.

SBP_SAMPLE_COLLISION

Statistical profiling samples collided .

SBP_SAMPLE_DISCARDED

Fields:

BUFFER_POINTER unsigned int
: Value of buffer pointer

DATA unsigned int
: Data discarded

SIZE unsigned int
: Size of sample, in bytes

SBP_SAMPLE_END

Statistical profiling of an operation ended.

SBP_SAMPLE_POP

Operation which might be sampled by statistical profiling.

SBP_SAMPLE_RECORDED

Fields:

BUFFER_ADDRESS unsigned int
: Virtual Address of start of buffer write

DATA unsigned int
: Data written to the buffer

SIZE unsigned int
: Size of sample written to buffer, in bytes

SBP_SAMPLE_START

Statistical profiling of an operation started.

SBP_SAMPLE_TAKEN

Operation sampled by statistical profiling after filtering.

SEMIHOSTING_CALL

Call of a semihost function occurred. Fields:

PC unsigned int
: The program counter after the semihosting call.

SEMIHOSTING_PRECALL

About to call semihost. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

EL2 bool
: Use EL2 translation regime.

NS bool
: Is Non-Secure.

REG_WIDTH unsigned int
: The current register width in bytes.

SIGNAL

External signal state change. Fields:

SIGNAL enum
: Signal that changed

STATE bool
: Signal asserted state

SOFTWARE_STEP

Return the Debug Software Step state. Fields:

STATE enum
: Software Step state

SPSR

SPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: Which of the banked SPSR registers is written.

OLD_VALUE unsigned int
: The old SPSR value

VALUE unsigned int
: The new SPSR value

START_COMPILE

Compilation started. Fields:

VADDR unsigned int
: Instruction where compilation begins.

STG_MULTIPLE_MICRO_OPS_ADDRESS_RANGE

Address range for MTE stores with multiple uops. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

STORE_MULTIPLE_REGS_ADDRESS_RANGE

Address range for stores with multiple registers. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

SUBOPTIMAL_LDST_RETIRED

suboptimal load/store. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1: half-word, 2: word ...)

OPTIMAL_ALIGN_SELECTOR unsigned int
: Selector to decide the optimal alignment size (1: access width, 2: 4-bytes, 3: 8-bytes, ..., 12: 4-kbytes)

VADDR unsigned int
: The virtual address of the access

SYNC

Called for every synchronization, that is every quantum break. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

LOCAL_QUANTUM unsigned int
: The local quantum of this CPU.

LOCAL_TIME unsigned int
: The local time of this CPU.

SYNC_REASON enum
: Reason for sync.

SYSCALL

System call instruction executed. Fields:

IMM unsigned int
: Immediate value of the system call instruction.

TYPE enum
: System call type.

VADDR unsigned int
: Instruction that caused the system call.

SYSREG_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

opc2 unsigned int
: opcode 2

SYSREG_UPDATE32

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE64

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

UNALIGNED_LDST_RETIRED

Processor unaligned load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1=>half-word, 2->word ...)

NUMBER_OF_BEATS unsigned int
: Number of accesses (beats) in this burst (Total size in bytes is ACCESS_SIZE * NUMBER_OF_BEATS)

VADDR unsigned int
: The virtual address of the access

VFP_D_REGS

VFP/NEON D 64 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_Q_REGS

VFP/NEON Q 128 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MASK unsigned int
: Each bit indicates a write to the corresponding byte in the register. For example, 0x1 would indicate a write to VALUE[7:0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_SYS_REGS

Writes to the VFP/NEON units system registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: Which VFP system register is written.

MASK unsigned int
: The mask indicating updated cumulative bits.

OLD_VALUE unsigned int
: The register's old value overwritten.

VALUE unsigned int
: The new value written to the VFP system register.

VFP_S_REGS

VFP/NEON S 32 bit register write. Fields:

ALIASING enum
: Type of register aliasing used.

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

WAYPOINT

Signals end of basic block execution. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: Origin instructions set

IS_COND bool
: Indicates if this is a conditional waypoint.

PC unsigned int
: Origin address (or 0 if unavailable)

TAKEN bool
: Indicates if this waypoint was taken.

TARGET unsigned int
: Destination address

TARGET_ISET enum
: Destination instructions set

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

WFE_END

WFE ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFE.

WFE_EVENT_REGISTER

WFE event register status: set/clear, reason. Fields:

INST_COUNT unsigned int
: Ticks count.

REASON enum
: Reason for set/clear. Only REASON==0 (Cleared by WFE) clears the bit, all other reasons set it.

WFE_IGNORED

WFE ignored. Fields:

EVENT bool
: This WFE was ignored because the event register was set.

INST_COUNT unsigned int
: Ticks count when ignoring WFE.

TRAPPED bool
: This WFE was trapped.

WFE_START

WFE entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFE.

WFI_END

WFI ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFI.

WFI_IGNORED

WFI ignored. Fields:

DISABLED bool
: This WFI was ignored because WFI is disabled.

INST_COUNT unsigned int
: Ticks count when ignoring WFI.

REASON enum
: specifies reason why WFI trace was ignored

TRAPPED bool
: This WFI was trapped.

WFI_START

WFI entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFI.

WFI_WAKEUP

WFI wakeup. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: Ticks count when WFI wakeup occurred.

REASON enum
: Reason for wakeup.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.23 ARM_Cortex-A77 - trace

This section describes the trace sources.

ASYNC_MEMORY_FAULT

Context ID Register write. Fields:

FAULT unsigned int
: Fault status in ESR format

PADDR unsigned int
: Physical Address (or 0 if unavailable)

VADDR signed int
: Virtual Address (or 0 if unavailable)

ATOMIC_END_ACCESS

Bus trace access for atomics. Fields:

ACCESS_FAIL bool
: Memory access failed

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

LOAD_VALUE unsigned int
: Loaded values

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ATOMIC_START_ACCESS

Bus trace access for atomics. Fields:

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

NSDESC unsigned int
: The security state of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ArchMsg.Error.exit_code

Fields:

EXITCODE signed int
: Exit code to be returned

KIND string
: Component kind that invoked the exit code trace

REASON string
: Reason why the exit code trace was invoked (optional)

ArchMsg.Warning.IgnoredWarmResetRequest

Event fires when software attempts to request warm reset through write to deprecated DBGPRCR.CWRR.

ArchMsg.Warning.RAS_ErrselrUnpredictableState

ERRSELR is in unpredictable state due to value greater than number_of_error_records written to it.

ArchMsg.Warning.RAS_UnpredictableStateWrite

Fields:

REG_NAME string
: Name of register being written.

ArchMsg.Warning.TarmacReconstructor

Unexpected event happened during Tarmac reconstruction. Fields:

REASON string
: Reason

ArchMsg.Warning.Unpredictably Indexed PM Event Register

Fields:

IsDirect unsigned int
: Direct Access rather than indirect

N unsigned int
: Number of Accessible Registers

SEL unsigned int
: Selector

ArchMsg.Warning.branch_to_unaligned_address

Fields:

ADDR unsigned int
: Unaligned branch address

ArchMsg.Warning.cache_contents_unknown

A cache has been activated without being invalidated and may contain UNKNOWN content. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

REGIME unsigned int
: Current translation regime

SIDE unsigned int
: 0 d-side cache or unified cache, 1 i-side

ArchMsg.Warning.dcimvac_matches_watchpoint

Watchpoints matching an AArch32 DCIMVA are implementation defined.

ArchMsg.Warning.decode_fieldmismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_fracbitsnegative

VCVT Instruction is unpredictable with a negative number of fraction bits.

ArchMsg.Warning.decode_initblock

Instruction is unpredictable in an IT block.

ArchMsg.Warning.decode_initblocknotlast

Instruction is unpredictable in an IT block when not the last.

ArchMsg.Warning.decode_invalid_condition

Instruction is unpredictable when the condition is not AL.

ArchMsg.Warning.decode_invalidfieldcombination

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_invalidvalue

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_mem_hint_unallocated

There is no memory hint allocated to this bit pattern.

ArchMsg.Warning.decode_registermatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_registermismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registeroutofrange

Float point instruction is unpredictable indexing beyond the end of the register bank.

ArchMsg.Warning.decode_sbzsbo

Reserved bits in the instruction are not canonical.

ArchMsg.Warning.decode_transactiontoobig

Attempt to perform a memory transaction of more than 128 bytes.

ArchMsg.Warning.decode_unpred_other

Instruction is unpredictable.

ArchMsg.Warning.decode_unpreduseofpc

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13inreglist

Instruction is unpredictable with R13 in register-list.

ArchMsg.Warning.decode_useofd16orhigher

Attempt to use register D16 or higher where it is not allowed.

ArchMsg.Warning.decode_writebackandbaseinlist

Instruction is unpredictable with write back when the base register is in the transfer list.

ArchMsg.Warning.decode_zeroregistersinlist

Instruction is unpredictable when no registers are to be transfered.

ArchMsg.Warning.recursive_branch

An instruction is performing a branch that targets the same instruction. If this was intended then a WFI might be more effective. Fields:

PC unsigned int
: Address of instruction causing the branch

ArchMsg.Warning.recursive_exception

An instruction has generated an exception that targets the same instruction. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CPSR unsigned int
: Processor state

PC unsigned int
: Address of instruction causing the exception

TYPE enum
: Type of exception

ArchMsg.Warning.reentrant_vector_catch

Fields:

ADDR unsigned int
: vector address

ArchMsg.Warning.reserved_it_state

Fields:

ITSTATE unsigned int
: reserved IT state

ArchMsg.Warning.tlb_contents_unknown

Translation has been enabled without invalidating the TLB. Fields:

EL unsigned int
: Exception level of translation regime

INVALIDITY unsigned int
: Which TLB may be invalid (if TLBs are separate)

NON_SECURE unsigned int
: Security state of translation regime

VMID unsigned int
: Current VMID, if applicable

ArchMsg.Warning.unknown_DLR_DSPSR

Exiting debug state with unknown DLR or DSPSR.

ArchMsg.Warning.unknown_ELR_SPSR

Returning from debug exception with unknown ELR or SPSR.

ArchMsg.Warning.unpredictable_a32_breakpoint

Fields:

ADDR1 unsigned int
: Address

ADDR2 unsigned int
: Breakpoint address

BAS unsigned int
: BAS Field

IS_ADDRESS_MISMATCH_BREAKPOINT bool
: Is Mismatch breakpoint, not a match breakpoint

N unsigned int
: Breakpoint number

OFFSET unsigned int
: Indicates which halfword actually considering into the A32 instruction

ArchMsg.Warning.unpredictable_banked_register_access

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

R unsigned int
R bit
SYSM unsigned int
SYSm

ArchMsg.Warning.unpredictable_ccfail_undef

Fields:

OPCODE unsigned int
opcode of the instruction
VADDR unsigned int
Virtual address of the instruction being executed

ArchMsg.Warning.unpredictable_ici_bits

ICI bits do not apply or are architecturally unpredictable for instruction.

ArchMsg.Warning.unpredictable_t32_breakpoint

Fields:

ADDR1 unsigned int
Address
ADDR2 unsigned int
Breakpoint address
BAS unsigned int
BAS Field
IS_ADDRESS_MISMATCH_BREAKPOINT bool
Is Mismatch breakpoint, not a match breakpoint
N unsigned int
Breakpoint number
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_t32_vector_catch

Fields:

ADDR unsigned int
Address
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_watchpoint_far

Fields:

LOWER_BOUND unsigned int
the lowest address accessed by the instruction that triggered the watchpoint
UPPER_BOUND unsigned int
the highest watchpointed address accessed by that instruction

ArchMsg.Warning.warning_AdvSIMDExpandImmUnexpectedZero

AdvSIMDExpandImm may treat this immediate value as UNPREDICTABLE.

ArchMsg.Warning.warning_ConditionalSMC

SMC instruction has UNPREDICTABLE effects when conditional when combined with traps.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_access_crosses_page_boundary

Access crosses page boundary.

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault_on_both_pages

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_spanning_different_memory

Fields:

ADDR unsigned int
: address of access crossing page

MEMTYPE_PAGE1 signed int
: First Page's memory type (0-Normal, 1-Non-Normal)

MEMTYPE_PAGE2 signed int
: Second Page's memory type (0-Normal, 1-Non-Normal)

ArchMsg.Warning.warning_access_wraps_around_memory

Fields:

WIDTH unsigned int
: Address width

ArchMsg.Warning.warning_bx_from_thumbee

Fields:

ADDR unsigned int
: Destination address

ArchMsg.Warning.warning_ccsidr_unimplemented_level

Fields:

CSSELR unsigned int
: current effective value of CSSELR

ArchMsg.Warning.warning_change_to_ns_when_tge_set

Attempting to change NS to 1 when HCR.TGE = 1.

ArchMsg.Warning.warning_contiguous_bit_check_abort

Fields:

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ArchMsg.Warning.warning_contiguous_bit_error

Fields:

CONTIG_ADDR unsigned int
: address of the TLB first read, with which this entry is expected to be contiguous

CONTIG_DATA unsigned int
: expected contents based on the entry at CONTIG_ADDR

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ENTRY_DATA unsigned int
: contents of conflicting TLB entry

ArchMsg.Warning.warning_cp10_cp11_mismatch

Fields:

EL unsigned int
: EL owning the control

ArchMsg.Warning.warning_cp10_cp11_reserved_value

Fields:

CP10 unsigned int
: CP10 access

CP11 unsigned int
: CP11 access

ArchMsg.Warning.warning_csselr_level_out_of_range

Fields:

IMPLEMENTED unsigned int
: number of cache levels implemented

LEVEL unsigned int
: written value of CSSELR.Level

TYPE enum
: The type of cache selected by CSSELR.InD and TnD

ArchMsg.Warning.warning_debug_flow_control_bits_not_obeyed

Fields:

ERROR enum
: type of error resulting

REGISTER enum
: register under consideration

ArchMsg.Warning.warning_debug_register_access_during_reset

Fields:

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

ArchMsg.Warning.warning_decode_cps_inconsistent_fields

The mode setting fields in the CPS instruction are inconsistent.

ArchMsg.Warning.warning_decode_invalid_state

Instruction is unpredictable in current exception-level/security state.

ArchMsg.Warning.warning_default_cacheable_mmu_on

Default cacheable is enabled (HCR.DC = 1) while MMU is enabled.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_default_cacheable_vmmu_off

Default cacheable is enabled (HCR.DC = 1) while HCR.VM = 0.

ArchMsg.Warning.warning_deprecated_wvr_bit_2_set

Fields:

N unsigned int
: index

WVR unsigned int
: DBGWVR

ArchMsg.Warning.warning_eret_while_software_step_active_pending

Missing ISB between setting MDSCR_EL1.SS and ERET.

ArchMsg.Warning.warning_exclusive_to_non_normal

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_exclusive_to_non_writeback

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_execute_from_device_memory

An attempt was made to execute from Device memory.

ArchMsg.Warning.warning_illegal_cpsr_mode

Fields:

NEW_MODE unsigned int
: New value of CPSR.M

ArchMsg.Warning.warning_illegal_srs_mode

Fields:

MODE unsigned int
: mode for Banked SP

ArchMsg.Warning.warning_implementation_defined_read_debug_register_in_SCS

Read Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_implementation_defined_sequential_security_transitions_supported

The behaviour of a sequential instruction fetches that cross from non-secure to secure memory and contain SG instruction is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_implementation_defined_stack_limit_check_supported

It is IMPLEMENTATION DEFINED whether stack pointer limit checking is performed for the this instructions. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDRESS unsigned int
: Address of instruction

OPCODE unsigned int
: instruction opcode

ArchMsg.Warning.warning_implementation_defined_write_debug_register_in_SCS

Write Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

DATA unsigned int
: data attempted to be written

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_invalid_tcr_granule

Fields:

REQUEST signed int
: page size requested (or 0 for reserved)

SUBSTITUTE signed int
: best guess available page size

TG_ID bool
: bits TG0 or TG1

ArchMsg.Warning.warning_load_multiple_user_registers_from_user_mode

An LDM(user registers) instruction executed from user mode.

ArchMsg.Warning.warning_load_pc_from_unaligned

PC loaded from an unaligned location.

ArchMsg.Warning.warning_mte_in_mem_attr_not_supported

Fields:

ATTR unsigned int
: Memory attribute

ArchMsg.Warning.warning_reserved_breakpoint_state_match

Fields:

IS_BREAKPOINT unsigned int
: Is breakpoint, not watchpoint

ArchMsg.Warning.warning_shareability

Unpredictable: combination of the 1st and 2nd stages of translation is Normal Inner Non-Cacheable, Outer Non-Cacheable.

ArchMsg.Warning.warning_software_step_set_while_enabled

MDSCR_EL1.SS set to 1 while software step debug exceptions are enabled.

ArchMsg.Warning.warning_thumb_instruction_wraps_around_memory

Thumb instruction warps around memory.

ArchMsg.Warning.warning_ttbr_sbz_bits_are_not_zero

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MSB unsigned int
x-1
TTBR enum
which TTBR register

ArchMsg.Warning.warning_unaligned_address_dbgdtrrx_write

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_address_dbgdtrtx_read

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_device

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_strongly_ordered

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unknown_sau_rnr

SAU_RNR was set to an unsupported value.

ArchMsg.Warning.warning_unpredictable_AIRCR_PRIP_and_BFHFNMINP

The effect of setting both AIRCR.BFHFNMINP and AIRCR.PRIP to 1 is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTCLRACTIVE_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTCLRACTIVE if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_and_SYSRESETREQ

When the processor is halted in Debug state, if a write to the register writes a 1 to both VECTRESET and SYSRESETREQ, the behavior is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTRESET if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_incorrect_VKEY

The value 0x05FA must be written to AIRCR.VKEY, otherwise the register write is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_EXE_RETURN_Reserved_Bit

EXC_RETURN[23:7] are reserved with the special condition that all bits should be written as one. Values other than all 1s are UNPREDICTABLE. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

EXC_RETURN unsigned int
: Special PC value

SBO_MASK unsigned int
: Which bits Should Be One to not UNPRED

ArchMsg.Warning.warning_unpredictable_change_to_DEMCR_MON_STEP_at_insufficient_priority

The effect of changing DEMCR.MON_STEP at an execution priority that is lower than the priority of the DebugMonitor exception is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_change_to_priority_of_active_exception

Changing the priority of an active exception is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_clear_lspact

Fields:

DRVEXC_TAKEN unsigned int
: derived exception taken not pended

REACHED_LAST_STORE unsigned int
: only the last faulted

ArchMsg.Warning.warning_unpredictable_dbg_exit_unaligned_dlr

Debug exit to AArch32 with DLR[0] set to 1. PC[0] can be set to 0 or to DLR[0].

ArchMsg.Warning.warning_unpredictable_exception_catch

Generation of an exception catch event is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_inconsistent_state

state on exception return is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_instruction

exception return instruction is unpredictable.

ArchMsg.Warning.warning_unpredictable_in_debug_state

Instruction is UNPREDICTABLE when executed in debug state.

ArchMsg.Warning.warning_unpredictable_mair_encoding

Fields:

ATTR unsigned int
: Memory attribute

ArchMsg.Warning.warning_unpredictable_pmu_counter_access

Access to PMU counters from non-secure EL0 or EL1 is UNPREDICTABLE with MDCR_EL2.HPMN set to 0.

ArchMsg.Warning.warning_unpredictable_prioritization_breakpoint_match_vector_catch

Two events with the same priority occurred at the same instruction: Address Matching Vector Catch debug event, and Breakpoint debug event It is constrained unpredictable which is taken.

ArchMsg.Warning.warning_unpredictable_stack_selection

UNPREDICTABLE state on context switch.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_tsize_out_of_range

Fields:

TCR enum
: Which TCR

TSIZE signed int
: Value of TSize

TSIZE_MAX signed int
: The maximum allowed value

TTBR unsigned int
: Which TTBR region

ArchMsg.Warning.warning_unpredictable_unaligned_pc_as_base_register

The PC value must be word-aligned, otherwise the behavior of the instruction is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_unaligned_pop_stack

Fields:

FRAME_ADDR unsigned int
: unaligned stack frame address

ArchMsg.Warning.warning_unpredictable_vmsa_memattrib

Fields:

TEXCB unsigned int
: texcb value

TEX_REMAP bool
: use tex remap

ArchMsg.Warning.warning_unpredictable_vtcr_t0sz_sl0

Fields:

SL0 signed int
: VTCR.SL0, starting level for stage 2 translation table walks

T0SZ signed int
: VTCR.T0SZ, required input address range

ArchMsg.Warning.warning_unpredictable_write_DHCSR

Fields:

NEW unsigned int
: data attempted to be written

OLD unsigned int
: previous value

ArchMsg.Warning.warning_unpredictable_write_SHCSR

UNPREDICTABLE setting a pending bit to 1 for an exception with priority >= execution priority.

ArchMsg.Warning.warning_unsupported_access_to_memory_mapped_register

Access to memory mapped register is unsupported.

ArchMsg.Warning.warning_user_jmcr_access

Fields:

WRITE unsigned int
: access is write

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_wcr_mask_and_bas

DBGWCR_EL1.MASK is non-zero and BAS is not 0xff.

ArchMsg.Warning.warning_wcr_mask_reserved

DBGWCR_ELn.MASK is set to a reserved value.

ArchMsg.Warning.warning_wcr_non_configuous_bas

DBGWCR_EL1.BAS has non-contiguous set of ones.

ArchMsg.Warning.warning_wvr_masked_bit_not_zero

DBGWVRn_EL1.VA contains a bit masked by DBGWCRn_EL1.MASK that is not zero.

ArchMsg.Why.why_branch_target_exception

A branch target exception has occurred. Fields:

BTYPE unsigned int
: pstate.BTYPE at time branch taken

OPCODE unsigned int
: opcode at branch target

PC unsigned int
: Virtual Address of branch target

ArchMsg.Why.why_illegal_state

An illegal mode change has occured. Fields:

MESSAGE string
: The message

BRANCH_MISPREDICT

Simulating branch mispredict. Fields:

PC unsigned int
: Origin address (or 0 if unavailable)

BRA_DIR

Direct branches, to immediate address. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

BRA_INDIR

Indirect branches, perhaps to a register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR_CCFAIL

Indirect branches fail, perhaps to a register.

BRA_RET_CCFAIL

Branche return fail, perhaps to a register.

CACHE_MAINTENANCE_OP

Cache Maintenance Operation. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Specified MVA or Set/way

FUNCTION enum
: Clean / Invalidate

SCOPE enum
: Affected region

SIDE enum
: Inst / Data.

CCFAIL

Conditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCFAIL_UNC

Unconditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS

Conditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS_UNC

Unconditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CHECKPOINT_MESSAGE

Report error messages from the checkpointing process. Fields:

message string
: Message contents.

CHECKPOINT_RESTORE_END

Checkpoint restore completed.

CHECKPOINT_RESTORE_START

Checkpoint restore about to start.

CHECKPOINT_SAVE_END

Checkpoint save completed.

CHECKPOINT_SAVE_START

Checkpoint save about to start.

CODE_CACHE_MAINT

Code cache maintenance. Fields:

END_INCL unsigned int
: Inclusive end addres for by VA or PA requests

NS bool
: Non-secure world for PA

START unsigned int
: Start address for by VA or PA requests

TYPE enum
: Request type.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

COMPILE_BLOCK_END

Last instruction of basic block translated. Fields:

VADDR unsigned int
: Address of next instruction after this basic block

COMPILE_BLOCK_START

First instruction of basic block translated. Fields:

VADDR unsigned int
: Address of first instruction in basic block

COMPILE_INST

ARM instruction compiled. Fields:

DISASS string
: Disassembly of the instruction.

ISET enum
: The instruction set of this instruction.

ITSTATE unsigned int
: The ITSTATE current for the instruction.

OPCODE unsigned int
: The opcode of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

CONTEXTIDR

Context ID Register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

NS bool
: Secure or nonsecure banked register is accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

CONTEXT_SYNC

Called for every context synchronization event. Fields:

ARCH_CSE bool
: Architectural Context Synchronization Event

SYSREG_SYNC bool
: System register synchronization event

CORE_ENDIAN

Core BE8 Big-Endian state changed. Fields:

BE8 bool
: Core BE8 Big-Endian state

CORE_INFO

Static processor attributes. Only triggered by a call to DumpState(). Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ARCH_PROFILE enum
: The architecture profile of the core.

CLUSTER_ID unsigned int
: The cluster ID of this processor.

CORE_NUM unsigned int
: The number of this core in an MP processor.

DETAILED_CORE_FEATURES enum
: Detailed core features as a bitfield. Enum values are bitmasks to extract the info from the field

FPU_VERSION enum
: The VFP version implemented by the core.

MEM_ARCH enum
: The memory architecture of the core.

MIDR_EL1 unsigned int
: MIDR_EL1 value

NUM_CORES unsigned int
: The number of cores in this MP processor.

QUANTUM_SIZE unsigned int
: The default quantum size of the core.

SECURITY_FEATURES bool
: Does the core have security features?

CORE_LOADS

Processor load accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CORE_REGS

Changes of the core registers R0 to R14. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number, 0 to 14.

MODE enum
: Bank of the register accessed.

OLD_VALUE unsigned int
: The old value overwritten.

PHYS_ID enum
: The physical register accessed.

VALUE unsigned int
: The new value written to the register.

CORE_REGS64

Changes of the core registers X0..X30, SP_ELn. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: The register number, 0..30 for X0..X30, >=32 for SP_ELn.

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

CORE_STORES

Processor store accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CP14_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP14_WRITE

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CP15_WRITE

System Control Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

CPSR

CPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

OLD_VALUE unsigned int
: The old CPSR value

UNKNOWN unsigned int
: Bits within the register that have unknown value.

VALUE unsigned int
: The new CPSR value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CRYPTO_SPEC

Every crypto instruction speculatively executed.

CorePowerStateChange

Core power-up state has changed. Fields:

PU bool
: Whether the core powered up

DATA_CACHE_SET_ALLOC_TAGS

Set allocation tags from DC GVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of memory in bytes.

TAG unsigned int
: The value of the tag being set.

VADDR unsigned int
: Virtual address used for setting allocation tags.

DATA_CACHE_ZERO

Zero and invalidate cache line from DC ZVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of zero'd memory in bytes.

VADDR unsigned int
: Virtual address of the zero'd cache line.

DEBUG_EVENT

Hardware debug support event. Fields:

EVENT enum
: Description of event

VALUE unsigned int
: data value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

END_COMPILE

Compilation end. Fields:

END_OF_PAGE_COUNT unsigned int
: Number of basic blocks exited due to page boundary since START_COMPILE.

FETCHFAIL_COUNT unsigned int
: Number of basic blocks exited due to ifetch failure START_COMPILE.

INST_COUNT unsigned int
: Number of instructions compiled since START_COMPILE.

NONSEQ_COUNT unsigned int
: Number of basic blocks exited due to non-sequential instructions since START_COMPILE.

PAGE_STRADDLE_COUNT unsigned int
: Number of basic blocks exited due to unaligned instructions crossing page since START_COMPILE.

ESB

Error Synchronization Barrier hit. Fields:

ASYNC bool
: ESB is async

IMPLICIT bool
: ESB is implicit

EXCEPTION

Exception taken. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

IS_PHYSICAL bool
: Physical or Virtual exception

LR unsigned int
: The value assigned to the link register.

PC unsigned int
: The location where the exception occurred.

PREFERRED_RETURN unsigned int
: The preferred return address for the exception.

TARGET_ISET enum
: The instruction set of the exception handler code.

TARGET_PC unsigned int
: The address the exception branches to.

VECTOR enum
: The exception vector.

VECTOR_OFFSET unsigned int
: The offset of an exception vector from the base address.

EXCEPTION_END

Every exception completed.

EXCEPTION_RAISE

Every exception raised.

EXCEPTION_RETURN

Branches on leaving exception. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

EXCEPTION_RETURN_PREBRANCH

Exception return event that must occur before PC and processor state have updated.

EXCEPTION_START

Every exception started.

FIQ_TAKEN

FIQ taken exception.

FP_STATE

Floating point state. Fields:

FPSR unsigned int
: FPSR register value.

MASK unsigned int
: The mask indicating updated cumulative bits.

HLT

HLT instruction occurred. Fields:

IMM16 unsigned int
: The 16-bit immediate value encoded in HLT instruction.

INST

Every instruction executed. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DEBUG_STATE bool
: The instruction is executed in debug state.

DISASS string
: Disassembly of instruction.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The current instruction set.

ITSTATE unsigned int
: The current ITSTATE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

MODE enum
: The mode the core is in.

NS unsigned int
: The core's non-secure bit.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

OPCODE unsigned int
: The opcode of the instruction.

PADDR unsigned int
: The physical address of the instruction.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

INST_END

Every instruction completed.

INST_START

Every instruction started. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

INST_STRADDLE

Instruction straddles boundary. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

IRQ_TAKEN

IRQ taken exception.

ISB

ISB instruction occurred.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOAD_MULTIPLE_REGS_LIST

List of destination registers (not source) of a multiple registers load instruction. Fields:

REG string
: Register to add into the list

LOCAL_MONITOR

Local monitor activity. Fields:

PADDR unsigned int
: Local Monitor Address

State enum
: State of the monitor (Open/Exclusive)

MEMMAP_DEBUG_READ

Memory mapped reads to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

MEMMAP_DEBUG_WRITE

Memory mapped writes to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

MMU_TRANS

Address translation information. Fields:

ASID unsigned int
: Address space identifier.

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Hyp bool
: Entry matches in Hyp state only

INNERCACHE_RA bool
: Is the inner cache allocate on read

INNERCACHE_TRANSIENT bool
: Is the inner write-through transient

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

MEMTYPE enum
: Memory type.

NSDESC enum
: Is secure side supposed to access secure or nonsecure memory

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TRANSIENT bool
: Is the outer write-through transient

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

PADDR unsigned int
: Physical address of the access.

PAGESIZE unsigned int
: Page size as log2(size).

SH enum
: Shareability

SIDE enum
: Inst / Data.

STAGE1_PERM unsigned int
: Stage 1 permissions, [5:3] Privileged access (XWR) [2:0] User access (XWR)

STAGE2_PERM unsigned int
: Stage 2 permission mask [5:3] Privileged access (XWR) [2:0] User access (XWR)

VADDR unsigned int
: Virtual address of the access.

VMID unsigned int
: Virtual machine identifier.

XS bool
: XS Attribute

nG enum
: Flag indicating whether ASID will be matched

MMU_TTB_READ

This event is triggered by reads caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data read.

IPA unsigned int
: For stage 1, the IPA of the read.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

OUTADDR unsigned int
: The output physical address.

PADDR unsigned int
: The physical address of the read.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MMU_TTB_WRITE

This event is triggered by writes caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data written.

IPA unsigned int
: For stage 1, the IPA of the write.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

PADDR unsigned int
: The physical address of the write.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MODE_CHANGE

Mode change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: The new mode.

NON_SECURE enum
: The new security state

OLD_MODE enum
: The old mode.

NOT_TAG_CHECKED_INST

MTE Tag not checked for current instruction.

PERIODIC

Called for every quantum. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

PC unsigned int
: The address of the next instruction to be executed on this CPU.

PMU_COUNTER_OVERFLOW

PMU counter overflow. Fields:

EVENT_ID unsigned int
: Event Identifier

INDEX unsigned int
: Counter index (as selected by PMSELR).

INTERRUPT bool
: Is interrupt enabled on overflow for this counter

PREFETCH_MEMORY64

Prefetch from PRFM or PRFUM instructions. Fields:

PRFOP unsigned int
: Prefetch hint

VADDR unsigned int
: Virtual address of the location that should be prefetched.

PRELOAD_DATA

Data preload from PLD instruction. Fields:

VADDR unsigned int
: Virtual address of the data that should be preloaded.

PRELOAD_INST

Instruction preload from PLI instruction. Fields:

VADDR unsigned int
: Virtual address of the instruction that should be preloaded.

PRE_CORE_LOAD

Trace just before a core load. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_CORE_STORE

Trace just before a core store. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_TTB_READ

This event is triggered before reads caused by a translation table walk. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR unsigned int
The physical address of the read.

PSTATE_PACTIVE_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Operating mode after the transition
OLD_STATE enum
Operating mode before the transition

PSTATE_PACTIVE_OPMODE_TRANSITION_DENIED

Trace the denied pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Operating mode after the transition
OLD_STATE enum
Operating mode before the transition
REASON string
Reason for the transition being denied

PSTATE_PACTIVE_POWER_TRANSITION

Trace pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Power mode after the transition
OLD_STATE enum
Power mode before the transition

PSTATE_PACTIVE_POWER_TRANSITION_DENIED

Trace the denied pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Power mode after the transition
OLD_STATE enum
Power mode before the transition
REASON string
Reason for the transition being denied

PSTATE_PACTIVE_THREAD_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions for threads on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Operating mode after the transition
OLD_STATE enum
Operating mode before the transition

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

RUN_STATE

Run state transition. Fields:

INST_COUNT unsigned int
: Ticks count at point of transition.

NEW enum
: New run state.

OLD enum
: Old run state.

SBP_SAMPLE_COLLISION

Statistical profiling samples collided .

SBP_SAMPLE_DISCARDED

Fields:

BUFFER_POINTER unsigned int
: Value of buffer pointer

DATA unsigned int
: Data discarded

SIZE unsigned int
: Size of sample, in bytes

SBP_SAMPLE_END

Statistical profiling of an operation ended.

SBP_SAMPLE_POP

Operation which might be sampled by statistical profiling.

SBP_SAMPLE_RECORDED

Fields:

BUFFER_ADDRESS unsigned int
: Virtual Address of start of buffer write

DATA unsigned int
: Data written to the buffer

SIZE unsigned int
: Size of sample written to buffer, in bytes

SBP_SAMPLE_START

Statistical profiling of an operation started.

SBP_SAMPLE_TAKEN

Operation sampled by statistical profiling after filtering.

SEMIHOSTING_CALL

Call of a semihost function occurred. Fields:

PC unsigned int
: The program counter after the semihosting call.

SEMIHOSTING_PRECALL

About to call semihost. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

EL2 bool
: Use EL2 translation regime.

NS bool
: Is Non-Secure.

REG_WIDTH unsigned int
: The current register width in bytes.

SIGNAL

External signal state change. Fields:

SIGNAL enum
: Signal that changed

STATE bool
: Signal asserted state

SOFTWARE_STEP

Return the Debug Software Step state. Fields:

STATE enum
: Software Step state

SPSR

SPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: Which of the banked SPSR registers is written.

OLD_VALUE unsigned int
: The old SPSR value

VALUE unsigned int
: The new SPSR value

START_COMPILE

Compilation started. Fields:

VADDR unsigned int
: Instruction where compilation begins.

STG_MULTIPLE_MICRO_OPS_ADDRESS_RANGE

Address range for MTE stores with multiple uops. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

STORE_MULTIPLE_REGS_ADDRESS_RANGE

Address range for stores with multiple registers. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

SUBOPTIMAL_LDST_RETIRED

suboptimal load/store. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1: half-word, 2: word ...)

OPTIMAL_ALIGN_SELECTOR unsigned int
: Selector to decide the optimal alignment size (1: access width, 2: 4-bytes, 3: 8-bytes, ..., 12: 4-kbytes)

VADDR unsigned int
: The virtual address of the access

SYNC

Called for every synchronization, that is every quantum break. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

LOCAL_QUANTUM unsigned int
: The local quantum of this CPU.

LOCAL_TIME unsigned int
: The local time of this CPU.

SYNC_REASON enum
: Reason for sync.

SYSCALL

System call instruction executed. Fields:

IMM unsigned int
: Immediate value of the system call instruction.

TYPE enum
: System call type.

VADDR unsigned int
: Instruction that caused the system call.

SYSREG_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

opc2 unsigned int
: opcode 2

SYSREG_UPDATE32

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE64

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

UNALIGNED_LDST_RETIRED

Processor unaligned load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1=>half-word, 2->word ...)

NUMBER_OF_BEATS unsigned int
: Number of accesses (beats) in this burst (Total size in bytes is ACCESS_SIZE * NUMBER_OF_BEATS)

VADDR unsigned int
: The virtual address of the access

VFP_D_REGS

VFP/NEON D 64 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_Q_REGS

VFP/NEON Q 128 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MASK unsigned int
: Each bit indicates a write to the corresponding byte in the register. For example, 0x1 would indicate a write to VALUE[7:0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_SYS_REGS

Writes to the VFP/NEON units system registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: Which VFP system register is written.

MASK unsigned int
: The mask indicating updated cumulative bits.

OLD_VALUE unsigned int
: The register's old value overwritten.

VALUE unsigned int
: The new value written to the VFP system register.

VFP_S_REGS

VFP/NEON S 32 bit register write. Fields:

ALIASING enum
: Type of register aliasing used.

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

WAYPOINT

Signals end of basic block execution. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: Origin instructions set

IS_COND bool
: Indicates if this is a conditional waypoint.

PC unsigned int
: Origin address (or 0 if unavailable)

TAKEN bool
: Indicates if this waypoint was taken.

TARGET unsigned int
: Destination address

TARGET_ISET enum
: Destination instructions set

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

WFE_END

WFE ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFE.

WFE_EVENT_REGISTER

WFE event register status: set/clear, reason. Fields:

INST_COUNT unsigned int
: Ticks count.

REASON enum
: Reason for set/clear. Only REASON==0 (Cleared by WFE) clears the bit, all other reasons set it.

WFE_IGNORED

WFE ignored. Fields:

EVENT bool
: This WFE was ignored because the event register was set.

INST_COUNT unsigned int
: Ticks count when ignoring WFE.

TRAPPED bool
: This WFE was trapped.

WFE_START

WFE entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFE.

WFI_END

WFI ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFI.

WFI_IGNORED

WFI ignored. Fields:

DISABLED bool
: This WFI was ignored because WFI is disabled.

INST_COUNT unsigned int
: Ticks count when ignoring WFI.

REASON enum
: specifies reason why WFI trace was ignored

TRAPPED bool
: This WFI was trapped.

WFI_START

WFI entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFI.

WFI_WAKEUP

WFI wakeup. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT `unsigned int`
: Ticks count when WFI wakeup occurred.

REASON `enum`
: Reason for wakeup.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.24 ARM_Cortex-A78 - trace

This section describes the trace sources.

ASYNC_MEMORY_FAULT

Context ID Register write. Fields:

FAULT unsigned int
: Fault status in ESR format

PADDR unsigned int
: Physical Address (or 0 if unavailable)

VADDR signed int
: Virtual Address (or 0 if unavailable)

ATOMIC_END_ACCESS

Bus trace access for atomics. Fields:

ACCESS_FAIL bool
: Memory access failed

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

LOAD_VALUE unsigned int
: Loaded values

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ATOMIC_START_ACCESS

Bus trace access for atomics. Fields:

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

NSDESC unsigned int
: The security state of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ArchMsg.Error.exit_code

Fields:

EXITCODE signed int
: Exit code to be returned

KIND string
: Component kind that invoked the exit code trace

REASON string
: Reason why the exit code trace was invoked (optional)

ArchMsg.Warning.IgnoredWarmResetRequest

Event fires when software attempts to request warm reset through write to deprecated DBGPRCR.CWRR.

ArchMsg.Warning.RAS_ErrselrUnpredictableState

ERRSELR is in unpredictable state due to value greater than number_of_error_records written to it.

ArchMsg.Warning.RAS_UnpredictableStateWrite

Fields:

REG_NAME string
: Name of register being written.

ArchMsg.Warning.TarmacReconstructor

Unexpected event happened during Tarmac reconstruction. Fields:

REASON string
: Reason

ArchMsg.Warning.Unpredictably Indexed PM Event Register

Fields:

IsDirect unsigned int
: Direct Access rather than indirect

N unsigned int
: Number of Accessible Registers

SEL unsigned int
: Selector

ArchMsg.Warning.branch_to_unaligned_address

Fields:

ADDR unsigned int
: Unaligned branch address

ArchMsg.Warning.cache_contents_unknown

A cache has been activated without being invalidated and may contain UNKNOWN content. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

REGIME unsigned int
: Current translation regime

SIDE unsigned int
: 0 d-side cache or unified cache, 1 i-side

ArchMsg.Warning.dcimvac_matches_watchpoint

Watchpoints matching an AArch32 DCIMVA are implementation defined.

ArchMsg.Warning.decode_fieldmismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_fracbitsnegative

VCVT Instruction is unpredictable with a negative number of fraction bits.

ArchMsg.Warning.decode_initblock

Instruction is unpredictable in an IT block.

ArchMsg.Warning.decode_initblocknotlast

Instruction is unpredictable in an IT block when not the last.

ArchMsg.Warning.decode_invalid_condition

Instruction is unpredictable when the condition is not AL.

ArchMsg.Warning.decode_invalidfieldcombination

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_invalidvalue

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_mem_hint_unallocated

There is no memory hint allocated to this bit pattern.

ArchMsg.Warning.decode_registermatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_registermismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registeroutofrange

Float point instruction is unpredictable indexing beyond the end of the register bank.

ArchMsg.Warning.decode_sbzsbo

Reserved bits in the instruction are not canonical.

ArchMsg.Warning.decode_transactiontoobig

Attempt to perform a memory transaction of more than 128 bytes.

ArchMsg.Warning.decode_unpred_other

Instruction is unpredictable.

ArchMsg.Warning.decode_unpreduseofpc

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13inreglist

Instruction is unpredictable with R13 in register-list.

ArchMsg.Warning.decode_useofd16orhigher

Attempt to use register D16 or higher where it is not allowed.

ArchMsg.Warning.decode_writebackandbaseinlist

Instruction is unpredictable with write back when the base register is in the transfer list.

ArchMsg.Warning.decode_zeroregistersinlist

Instruction is unpredictable when no registers are to be transfered.

ArchMsg.Warning.recursive_branch

An instruction is performing a branch that targets the same instruction. If this was intended then a WFI might be more effective. Fields:

PC unsigned int
: Address of instruction causing the branch

ArchMsg.Warning.recursive_exception

An instruction has generated an exception that targets the same instruction. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CPSR unsigned int
: Processor state

PC unsigned int
: Address of instruction causing the exception

TYPE enum
: Type of exception

ArchMsg.Warning.reentrant_vector_catch

Fields:

ADDR unsigned int
: vector address

ArchMsg.Warning.reserved_it_state

Fields:

ITSTATE unsigned int
: reserved IT state

ArchMsg.Warning.tlb_contents_unknown

Translation has been enabled without invalidating the TLB. Fields:

EL unsigned int
: Exception level of translation regime

INVALIDITY unsigned int
: Which TLB may be invalid (if TLBs are separate)

NON_SECURE unsigned int
: Security state of translation regime

VMID unsigned int
: Current VMID, if applicable

ArchMsg.Warning.unknown_DLR_DSPSR

Exiting debug state with unknown DLR or DSPSR.

ArchMsg.Warning.unknown_ELR_SPSR

Returning from debug exception with unknown ELR or SPSR.

ArchMsg.Warning.unpredictable_a32_breakpoint

Fields:

ADDR1 unsigned int
: Address

ADDR2 unsigned int
: Breakpoint address

BAS unsigned int
: BAS Field

IS_ADDRESS_MISMATCH_BREAKPOINT bool
: Is Mismatch breakpoint, not a match breakpoint

N unsigned int
: Breakpoint number

OFFSET unsigned int
: Indicates which halfword actually considering into the A32 instruction

ArchMsg.Warning.unpredictable_banked_register_access

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

R unsigned int
R bit
SYSM unsigned int
SYSm

ArchMsg.Warning.unpredictable_ccfail_undef

Fields:

OPCODE unsigned int
opcode of the instruction
VADDR unsigned int
Virtual address of the instruction being executed

ArchMsg.Warning.unpredictable_ici_bits

ICI bits do not apply or are architecturally unpredictable for instruction.

ArchMsg.Warning.unpredictable_t32_breakpoint

Fields:

ADDR1 unsigned int
Address
ADDR2 unsigned int
Breakpoint address
BAS unsigned int
BAS Field
IS_ADDRESS_MISMATCH_BREAKPOINT bool
Is Mismatch breakpoint, not a match breakpoint
N unsigned int
Breakpoint number
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_t32_vector_catch

Fields:

ADDR unsigned int
Address
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_watchpoint_far

Fields:

LOWER_BOUND unsigned int
the lowest address accessed by the instruction that triggered the watchpoint
UPPER_BOUND unsigned int
the highest watchpointed address accessed by that instruction

ArchMsg.Warning.warning_AdvSIMDExpandImmUnexpectedZero

AdvSIMDExpandImm may treat this immediate value as UNPREDICTABLE.

ArchMsg.Warning.warning_ConditionalSMC

SMC instruction has UNPREDICTABLE effects when conditional when combined with traps.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_access_crosses_page_boundary

Access crosses page boundary.

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault_on_both_pages

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_spanning_different_memory

Fields:

ADDR unsigned int
: address of access crossing page

MEMTYPE_PAGE1 signed int
: First Page's memory type (0-Normal, 1-Non-Normal)

MEMTYPE_PAGE2 signed int
: Second Page's memory type (0-Normal, 1-Non-Normal)

ArchMsg.Warning.warning_access_wraps_around_memory

Fields:

WIDTH unsigned int
: Address width

ArchMsg.Warning.warning_bx_from_thumbee

Fields:

ADDR unsigned int
: Destination address

ArchMsg.Warning.warning_ccsidr_unimplemented_level

Fields:

CSSELR unsigned int
: current effective value of CSSELR

ArchMsg.Warning.warning_change_to_ns_when_tge_set

Attempting to change NS to 1 when HCR.TGE = 1.

ArchMsg.Warning.warning_contiguous_bit_check_abort

Fields:

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ArchMsg.Warning.warning_contiguous_bit_error

Fields:

CONTIG_ADDR unsigned int
: address of the TLB first read, with which this entry is expected to be contiguous

CONTIG_DATA unsigned int
: expected contents based on the entry at CONTIG_ADDR

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ENTRY_DATA unsigned int
: contents of conflicting TLB entry

ArchMsg.Warning.warning_cp10_cp11_mismatch

Fields:

EL unsigned int
: EL owning the control

ArchMsg.Warning.warning_cp10_cp11_reserved_value

Fields:

CP10 unsigned int
: CP10 access

CP11 unsigned int
: CP11 access

ArchMsg.Warning.warning_csselr_level_out_of_range

Fields:

IMPLEMENTED unsigned int
: number of cache levels implemented

LEVEL unsigned int
: written value of CSSELR.Level

TYPE enum
: The type of cache selected by CSSELR.InD and TnD

ArchMsg.Warning.warning_debug_flow_control_bits_not_obeyed

Fields:

ERROR enum
: type of error resulting

REGISTER enum
: register under consideration

ArchMsg.Warning.warning_debug_register_access_during_reset

Fields:

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

ArchMsg.Warning.warning_decode_cps_inconsistent_fields

The mode setting fields in the CPS instruction are inconsistent.

ArchMsg.Warning.warning_decode_invalid_state

Instruction is unpredictable in current exception-level/security state.

ArchMsg.Warning.warning_default_cacheable_mmu_on

Default cacheable is enabled (HCR.DC = 1) while MMU is enabled.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_default_cacheable_vmmu_off

Default cacheable is enabled (HCR.DC = 1) while HCR.VM = 0.

ArchMsg.Warning.warning_deprecated_wvr_bit_2_set

Fields:

N unsigned int
: index

WVR unsigned int
: DBGWVR

ArchMsg.Warning.warning_eret_while_software_step_active_pending

Missing ISB between setting MDSCR_EL1.SS and ERET.

ArchMsg.Warning.warning_exclusive_to_non_normal

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_exclusive_to_non_writeback

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_execute_from_device_memory

An attempt was made to execute from Device memory.

ArchMsg.Warning.warning_illegal_cpsr_mode

Fields:

NEW_MODE unsigned int
: New value of CPSR.M

ArchMsg.Warning.warning_illegal_srs_mode

Fields:

MODE unsigned int
: mode for Banked SP

ArchMsg.Warning.warning_implementation_defined_read_debug_register_in_SCS

Read Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_implementation_defined_sequential_security_transitions_supported

The behaviour of a sequential instruction fetches that cross from non-secure to secure memory and contain SG instruction is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_implementation_defined_stack_limit_check_supported

It is IMPLEMENTATION DEFINED whether stack pointer limit checking is performed for the this instructions. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDRESS unsigned int
: Address of instruction

OPCODE unsigned int
: instruction opcode

ArchMsg.Warning.warning_implementation_defined_write_debug_register_in_SCS

Write Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

DATA unsigned int
: data attempted to be written

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_invalid_tcr_granule

Fields:

REQUEST signed int
: page size requested (or 0 for reserved)

SUBSTITUTE signed int
: best guess available page size

TG_ID bool
: bits TG0 or TG1

ArchMsg.Warning.warning_load_multiple_user_registers_from_user_mode

An LDM(user registers) instruction executed from user mode.

ArchMsg.Warning.warning_load_pc_from_unaligned

PC loaded from an unaligned location.

ArchMsg.Warning.warning_mte_in_mem_attr_not_supported

Fields:

ATTR unsigned int
: Memory attribute

ArchMsg.Warning.warning_reserved_breakpoint_state_match

Fields:

IS_BREAKPOINT unsigned int
: Is breakpoint, not watchpoint

ArchMsg.Warning.warning_shareability

Unpredictable: combination of the 1st and 2nd stages of translation is Normal Inner Non-Cacheable, Outer Non-Cacheable.

ArchMsg.Warning.warning_software_step_set_while_enabled

MDSCR_EL1.SS set to 1 while software step debug exceptions are enabled.

ArchMsg.Warning.warning_thumb_instruction_wraps_around_memory

Thumb instruction warps around memory.

ArchMsg.Warning.warning_ttbr_sbz_bits_are_not_zero

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MSB unsigned int
x-1
TTBR enum
which TTBR register

ArchMsg.Warning.warning_unaligned_address_dbgdtrrx_write

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_address_dbgdtrtx_read

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_device

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_strongly_ordered

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unknown_sau_rnr

SAU_RNR was set to an unsupported value.

ArchMsg.Warning.warning_unpredictable_AIRCR_PRIP_and_BFHFNMINP

The effect of setting both AIRCR.BFHFNMINP and AIRCR.PRIP to 1 is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTCLRACTIVE_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTCLRACTIVE if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_and_SYSRESETREQ

When the processor is halted in Debug state, if a write to the register writes a 1 to both VECTRESET and SYSRESETREQ, the behavior is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTRESET if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_incorrect_VKEY

The value 0x05FA must be written to AIRCR.VKEY, otherwise the register write is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_EXE_RETURN_Reserved_Bit

EXC_RETURN[23:7] are reserved with the special condition that all bits should be written as one. Values other than all 1s are UNPREDICTABLE. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

EXC_RETURN unsigned int
: Special PC value

SBO_MASK unsigned int
: Which bits Should Be One to not UNPRED

ArchMsg.Warning.warning_unpredictable_change_to_DEMCR_MON_STEP_at_insufficient_priority

The effect of changing DEMCR.MON_STEP at an execution priority that is lower than the priority of the DebugMonitor exception is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_change_to_priority_of_active_exception

Changing the priority of an active exception is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_clear_lspact

Fields:

DRVEXC_TAKEN unsigned int
: derived exception taken not pended

REACHED_LAST_STORE unsigned int
: only the last faulted

ArchMsg.Warning.warning_unpredictable_dbg_exit_unaligned_dlr

Debug exit to AArch32 with DLR[0] set to 1. PC[0] can be set to 0 or to DLR[0].

ArchMsg.Warning.warning_unpredictable_exception_catch

Generation of an exception catch event is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_inconsistent_state

state on exception return is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_instruction

exception return instruction is unpredictable.

ArchMsg.Warning.warning_unpredictable_in_debug_state

Instruction is UNPREDICTABLE when executed in debug state.

ArchMsg.Warning.warning_unpredictable_mair_encoding

Fields:

ATTR unsigned int
: Memory attribute

ArchMsg.Warning.warning_unpredictable_pmu_counter_access

Access to PMU counters from non-secure EL0 or EL1 is UNPREDICTABLE with MDCR_EL2.HPMN set to 0.

ArchMsg.Warning.warning_unpredictable_prioritization_breakpoint_match_vector_catch

Two events with the same priority occurred at the same instruction: Address Matching Vector Catch debug event, and Breakpoint debug event It is constrained unpredictable which is taken.

ArchMsg.Warning.warning_unpredictable_stack_selection

UNPREDICTABLE state on context switch.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_tsize_out_of_range

Fields:

TCR enum
: Which TCR

TSIZE signed int
: Value of TSize

TSIZE_MAX signed int
: The maximum allowed value

TTBR unsigned int
: Which TTBR region

ArchMsg.Warning.warning_unpredictable_unaligned_pc_as_base_register

The PC value must be word-aligned, otherwise the behavior of the instruction is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_unaligned_pop_stack

Fields:

FRAME_ADDR unsigned int
: unaligned stack frame address

ArchMsg.Warning.warning_unpredictable_vmsa_memattrib

Fields:

TEXCB unsigned int
: texcb value

TEX_REMAP bool
: use tex remap

ArchMsg.Warning.warning_unpredictable_vtcr_t0sz_sl0

Fields:

SL0 signed int
: VTCR.SL0, starting level for stage 2 translation table walks

T0SZ signed int
: VTCR.T0SZ, required input address range

ArchMsg.Warning.warning_unpredictable_write_DHCSR

Fields:

NEW unsigned int
: data attempted to be written

OLD unsigned int
: previous value

ArchMsg.Warning.warning_unpredictable_write_SHCSR

UNPREDICTABLE setting a pending bit to 1 for an exception with priority >= execution priority.

ArchMsg.Warning.warning_unsupported_access_to_memory_mapped_register

Access to memory mapped register is unsupported.

ArchMsg.Warning.warning_user_jmcr_access

Fields:

WRITE unsigned int
: access is write

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_wcr_mask_and_bas

DBGWCR_EL1.MASK is non-zero and BAS is not 0xff.

ArchMsg.Warning.warning_wcr_mask_reserved

DBGWCR_ELn.MASK is set to a reserved value.

ArchMsg.Warning.warning_wcr_non_configuous_bas

DBGWCR_EL1.BAS has non-contiguous set of ones.

ArchMsg.Warning.warning_wvr_masked_bit_not_zero

DBGWVRn_EL1.VA contains a bit masked by DBGWCRn_EL1.MASK that is not zero.

ArchMsg.Why.why_branch_target_exception

A branch target exception has occurred. Fields:

BTYPE unsigned int
: pstate.BTYPE at time branch taken

OPCODE unsigned int
: opcode at branch target

PC unsigned int
: Virtual Address of branch target

ArchMsg.Why.why_illegal_state

An illegal mode change has occured. Fields:

MESSAGE string
: The message

BRANCH_MISPREDICT

Simulating branch mispredict. Fields:

PC unsigned int
: Origin address (or 0 if unavailable)

BRA_DIR

Direct branches, to immediate address. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

BRA_INDIR

Indirect branches, perhaps to a register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR_CCFAIL

Indirect branches fail, perhaps to a register.

BRA_RET_CCFAIL

Branche return fail, perhaps to a register.

CACHE_MAINTENANCE_OP

Cache Maintenance Operation. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Specified MVA or Set/way

FUNCTION enum
: Clean / Invalidate

SCOPE enum
: Affected region

SIDE enum
: Inst / Data.

CCFAIL

Conditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCFAIL_UNC

Unconditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS

Conditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS_UNC

Unconditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CHECKPOINT_MESSAGE

Report error messages from the checkpointing process. Fields:

message string
: Message contents.

CHECKPOINT_RESTORE_END

Checkpoint restore completed.

CHECKPOINT_RESTORE_START

Checkpoint restore about to start.

CHECKPOINT_SAVE_END

Checkpoint save completed.

CHECKPOINT_SAVE_START

Checkpoint save about to start.

CODE_CACHE_MAINT

Code cache maintenance. Fields:

END_INCL unsigned int
: Inclusive end addres for by VA or PA requests

NS bool
: Non-secure world for PA

START unsigned int
: Start address for by VA or PA requests

TYPE enum
: Request type.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

COMPILE_BLOCK_END

Last instruction of basic block translated. Fields:

VADDR unsigned int
: Address of next instruction after this basic block

COMPILE_BLOCK_START

First instruction of basic block translated. Fields:

VADDR unsigned int
: Address of first instruction in basic block

COMPILE_INST

ARM instruction compiled. Fields:

DISASS string
: Disassembly of the instruction.

ISET enum
: The instruction set of this instruction.

ITSTATE unsigned int
: The ITSTATE current for the instruction.

OPCODE unsigned int
: The opcode of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

CONTEXTIDR

Context ID Register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

NS bool
: Secure or nonsecure banked register is accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

CONTEXT_SYNC

Called for every context synchronization event. Fields:

ARCH_CSE bool
: Architectural Context Synchronization Event

SYSREG_SYNC bool
: System register synchronization event

CORE_ENDIAN

Core BE8 Big-Endian state changed. Fields:

BE8 bool
: Core BE8 Big-Endian state

CORE_INFO

Static processor attributes. Only triggered by a call to DumpState(). Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ARCH_PROFILE enum
: The architecture profile of the core.

CLUSTER_ID unsigned int
: The cluster ID of this processor.

CORE_NUM unsigned int
: The number of this core in an MP processor.

DETAILED_CORE_FEATURES enum
: Detailed core features as a bitfield. Enum values are bitmasks to extract the info from the field

FPU_VERSION enum
: The VFP version implemented by the core.

MEM_ARCH enum
: The memory architecture of the core.

MIDR_EL1 unsigned int
: MIDR_EL1 value

NUM_CORES unsigned int
: The number of cores in this MP processor.

QUANTUM_SIZE unsigned int
: The default quantum size of the core.

SECURITY_FEATURES bool
: Does the core have security features?

CORE_LOADS

Processor load accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CORE_REGS

Changes of the core registers R0 to R14. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number, 0 to 14.

MODE enum
: Bank of the register accessed.

OLD_VALUE unsigned int
: The old value overwritten.

PHYS_ID enum
: The physical register accessed.

VALUE unsigned int
: The new value written to the register.

CORE_REGS64

Changes of the core registers X0..X30, SP_ELn. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: The register number, 0..30 for X0..X30, >=32 for SP_ELn.

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

CORE_STORES

Processor store accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CP14_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP14_WRITE

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CP15_WRITE

System Control Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

CPSR

CPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

OLD_VALUE unsigned int
: The old CPSR value

UNKNOWN unsigned int
: Bits within the register that have unknown value.

VALUE unsigned int
: The new CPSR value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CRYPTO_SPEC

Every crypto instruction speculatively executed.

CorePowerStateChange

Core power-up state has changed. Fields:

PU bool
: Whether the core powered up

DATA_CACHE_SET_ALLOC_TAGS

Set allocation tags from DC GVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of memory in bytes.

TAG unsigned int
: The value of the tag being set.

VADDR unsigned int
: Virtual address used for setting allocation tags.

DATA_CACHE_ZERO

Zero and invalidate cache line from DC ZVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of zero'd memory in bytes.

VADDR unsigned int
: Virtual address of the zero'd cache line.

DEBUG_EVENT

Hardware debug support event. Fields:

EVENT enum
: Description of event

VALUE unsigned int
: data value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

END_COMPILE

Compilation end. Fields:

END_OF_PAGE_COUNT unsigned int
: Number of basic blocks exited due to page boundary since START_COMPILE.

FETCHFAIL_COUNT unsigned int
: Number of basic blocks exited due to ifetch failure START_COMPILE.

INST_COUNT unsigned int
: Number of instructions compiled since START_COMPILE.

NONSEQ_COUNT unsigned int
: Number of basic blocks exited due to non-sequential instructions since START_COMPILE.

PAGE_STRADDLE_COUNT unsigned int
: Number of basic blocks exited due to unaligned instructions crossing page since START_COMPILE.

ESB

Error Synchronization Barrier hit. Fields:

ASYNC bool
: ESB is async

IMPLICIT bool
: ESB is implicit

EXCEPTION

Exception taken. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

IS_PHYSICAL bool
: Physical or Virtual exception

LR unsigned int
: The value assigned to the link register.

PC unsigned int
: The location where the exception occurred.

PREFERRED_RETURN unsigned int
: The preferred return address for the exception.

TARGET_ISET enum
: The instruction set of the exception handler code.

TARGET_PC unsigned int
: The address the exception branches to.

VECTOR enum
: The exception vector.

VECTOR_OFFSET unsigned int
: The offset of an exception vector from the base address.

EXCEPTION_END

Every exception completed.

EXCEPTION_RAISE

Every exception raised.

EXCEPTION_RETURN

Branches on leaving exception. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

EXCEPTION_RETURN_PREBRANCH

Exception return event that must occur before PC and processor state have updated.

EXCEPTION_START

Every exception started.

FIQ_TAKEN

FIQ taken exception.

FP_STATE

Floating point state. Fields:

FPSR unsigned int
: FPSR register value.

MASK unsigned int
: The mask indicating updated cumulative bits.

HLT

HLT instruction occurred. Fields:

IMM16 unsigned int
: The 16-bit immediate value encoded in HLT instruction.

INST

Every instruction executed. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DEBUG_STATE bool
: The instruction is executed in debug state.

DISASS string
: Disassembly of instruction.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The current instruction set.

ITSTATE unsigned int
: The current ITSTATE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

MODE enum
: The mode the core is in.

NS unsigned int
: The core's non-secure bit.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

OPCODE unsigned int
: The opcode of the instruction.

PADDR unsigned int
: The physical address of the instruction.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

INST_END

Every instruction completed.

INST_START

Every instruction started. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

INST_STRADDLE

Instruction straddles boundary. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

IRQ_TAKEN

IRQ taken exception.

ISB

ISB instruction occurred.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOAD_MULTIPLE_REGS_LIST

List of destination registers (not source) of a multiple registers load instruction. Fields:

REG string
: Register to add into the list

LOCAL_MONITOR

Local monitor activity. Fields:

PADDR unsigned int
: Local Monitor Address

State enum
: State of the monitor (Open/Exclusive)

MEMMAP_DEBUG_READ

Memory mapped reads to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

MEMMAP_DEBUG_WRITE

Memory mapped writes to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

MMU_TRANS

Address translation information. Fields:

ASID unsigned int
: Address space identifier.

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Hyp bool
: Entry matches in Hyp state only

INNERCACHE_RA bool
: Is the inner cache allocate on read

INNERCACHE_TRANSIENT bool
: Is the inner write-through transient

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

MEMTYPE enum
: Memory type.

NSDESC enum
: Is secure side supposed to access secure or nonsecure memory

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TRANSIENT bool
: Is the outer write-through transient

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

PADDR unsigned int
: Physical address of the access.

PAGESIZE unsigned int
: Page size as log2(size).

SH enum
: Shareability

SIDE enum
: Inst / Data.

STAGE1_PERM unsigned int
: Stage 1 permissions, [5:3] Privileged access (XWR) [2:0] User access (XWR)

STAGE2_PERM unsigned int
: Stage 2 permission mask [5:3] Privileged access (XWR) [2:0] User access (XWR)

VADDR unsigned int
: Virtual address of the access.

VMID unsigned int
: Virtual machine identifier.

XS bool
: XS Attribute

nG enum
: Flag indicating whether ASID will be matched

MMU_TTB_READ

This event is triggered by reads caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data read.

IPA unsigned int
: For stage 1, the IPA of the read.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

OUTADDR unsigned int
: The output physical address.

PADDR unsigned int
: The physical address of the read.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MMU_TTB_WRITE

This event is triggered by writes caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data written.

IPA unsigned int
: For stage 1, the IPA of the write.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

PADDR unsigned int
: The physical address of the write.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MODE_CHANGE

Mode change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: The new mode.

NON_SECURE enum
: The new security state

OLD_MODE enum
: The old mode.

NOT_TAG_CHECKED_INST

MTE Tag not checked for current instruction.

PERIODIC

Called for every quantum. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

PC unsigned int
: The address of the next instruction to be executed on this CPU.

PMU_COUNTER_OVERFLOW

PMU counter overflow. Fields:

EVENT_ID unsigned int
: Event Identifier

INDEX unsigned int
: Counter index (as selected by PMSELR).

INTERRUPT bool
: Is interrupt enabled on overflow for this counter

PREFETCH_MEMORY64

Prefetch from PRFM or PRFUM instructions. Fields:

PRFOP unsigned int
: Prefetch hint

VADDR unsigned int
: Virtual address of the location that should be prefetched.

PRELOAD_DATA

Data preload from PLD instruction. Fields:

VADDR unsigned int
: Virtual address of the data that should be preloaded.

PRELOAD_INST

Instruction preload from PLI instruction. Fields:

VADDR unsigned int
: Virtual address of the instruction that should be preloaded.

PRE_CORE_LOAD

Trace just before a core load. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_CORE_STORE

Trace just before a core store. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_TTB_READ

This event is triggered before reads caused by a translation table walk. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR unsigned int
The physical address of the read.

PSTATE_PACTIVE_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Operating mode after the transition
OLD_STATE enum
Operating mode before the transition

PSTATE_PACTIVE_OPMODE_TRANSITION_DENIED

Trace the denied pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Operating mode after the transition
OLD_STATE enum
Operating mode before the transition
REASON string
Reason for the transition being denied

PSTATE_PACTIVE_POWER_TRANSITION

Trace pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Power mode after the transition
OLD_STATE enum
Power mode before the transition

PSTATE_PACTIVE_POWER_TRANSITION_DENIED

Trace the denied pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Power mode after the transition
OLD_STATE enum
Power mode before the transition
REASON string
Reason for the transition being denied

PSTATE_PACTIVE_THREAD_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions for threads on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Operating mode after the transition
OLD_STATE enum
Operating mode before the transition

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

RUN_STATE

Run state transition. Fields:

INST_COUNT unsigned int
: Ticks count at point of transition.

NEW enum
: New run state.

OLD enum
: Old run state.

SBP_SAMPLE_COLLISION

Statistical profiling samples collided .

SBP_SAMPLE_DISCARDED

Fields:

BUFFER_POINTER unsigned int
: Value of buffer pointer

DATA unsigned int
: Data discarded

SIZE unsigned int
: Size of sample, in bytes

SBP_SAMPLE_END

Statistical profiling of an operation ended.

SBP_SAMPLE_POP

Operation which might be sampled by statistical profiling.

SBP_SAMPLE_RECORDED

Fields:

BUFFER_ADDRESS unsigned int
: Virtual Address of start of buffer write

DATA unsigned int
: Data written to the buffer

SIZE unsigned int
: Size of sample written to buffer, in bytes

SBP_SAMPLE_START

Statistical profiling of an operation started.

SBP_SAMPLE_TAKEN

Operation sampled by statistical profiling after filtering.

SEMIHOSTING_CALL

Call of a semihost function occurred. Fields:

PC unsigned int
: The program counter after the semihosting call.

SEMIHOSTING_PRECALL

About to call semihost. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

EL2 bool
: Use EL2 translation regime.

NS bool
: Is Non-Secure.

REG_WIDTH unsigned int
: The current register width in bytes.

SIGNAL

External signal state change. Fields:

SIGNAL enum
: Signal that changed

STATE bool
: Signal asserted state

SOFTWARE_STEP

Return the Debug Software Step state. Fields:

STATE enum
: Software Step state

SPSR

SPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: Which of the banked SPSR registers is written.

OLD_VALUE unsigned int
: The old SPSR value

VALUE unsigned int
: The new SPSR value

START_COMPILE

Compilation started. Fields:

VADDR unsigned int
: Instruction where compilation begins.

STG_MULTIPLE_MICRO_OPS_ADDRESS_RANGE

Address range for MTE stores with multiple uops. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

STORE_MULTIPLE_REGS_ADDRESS_RANGE

Address range for stores with multiple registers. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

SUBOPTIMAL_LDST_RETIRED

suboptimal load/store. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1: half-word, 2: word ...)

OPTIMAL_ALIGN_SELECTOR unsigned int
: Selector to decide the optimal alignment size (1: access width, 2: 4-bytes, 3: 8-bytes, ..., 12: 4-kbytes)

VADDR unsigned int
: The virtual address of the access

SYNC

Called for every synchronization, that is every quantum break. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

LOCAL_QUANTUM unsigned int
: The local quantum of this CPU.

LOCAL_TIME unsigned int
: The local time of this CPU.

SYNC_REASON enum
: Reason for sync.

SYSCALL

System call instruction executed. Fields:

IMM unsigned int
: Immediate value of the system call instruction.

TYPE enum
: System call type.

VADDR unsigned int
: Instruction that caused the system call.

SYSREG_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

opc2 unsigned int
: opcode 2

SYSREG_UPDATE32

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE64

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

UNALIGNED_LDST_RETIRED

Processor unaligned load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1=>half-word, 2->word ...)

NUMBER_OF_BEATS unsigned int
: Number of accesses (beats) in this burst (Total size in bytes is ACCESS_SIZE * NUMBER_OF_BEATS)

VADDR unsigned int
: The virtual address of the access

VFP_D_REGS

VFP/NEON D 64 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_Q_REGS

VFP/NEON Q 128 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MASK unsigned int
: Each bit indicates a write to the corresponding byte in the register. For example, 0x1 would indicate a write to VALUE[7:0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_SYS_REGS

Writes to the VFP/NEON units system registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: Which VFP system register is written.

MASK unsigned int
: The mask indicating updated cumulative bits.

OLD_VALUE unsigned int
: The register's old value overwritten.

VALUE unsigned int
: The new value written to the VFP system register.

VFP_S_REGS

VFP/NEON S 32 bit register write. Fields:

ALIASING enum
: Type of register aliasing used.

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

WAYPOINT

Signals end of basic block execution. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: Origin instructions set

IS_COND bool
: Indicates if this is a conditional waypoint.

PC unsigned int
: Origin address (or 0 if unavailable)

TAKEN bool
: Indicates if this waypoint was taken.

TARGET unsigned int
: Destination address

TARGET_ISET enum
: Destination instructions set

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

WFE_END

WFE ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFE.

WFE_EVENT_REGISTER

WFE event register status: set/clear, reason. Fields:

INST_COUNT unsigned int
: Ticks count.

REASON enum
: Reason for set/clear. Only REASON==0 (Cleared by WFE) clears the bit, all other reasons set it.

WFE_IGNORED

WFE ignored. Fields:

EVENT bool
: This WFE was ignored because the event register was set.

INST_COUNT unsigned int
: Ticks count when ignoring WFE.

TRAPPED bool
: This WFE was trapped.

WFE_START

WFE entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFE.

WFI_END

WFI ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFI.

WFI_IGNORED

WFI ignored. Fields:

DISABLED bool
: This WFI was ignored because WFI is disabled.

INST_COUNT unsigned int
: Ticks count when ignoring WFI.

REASON enum
: specifies reason why WFI trace was ignored

TRAPPED bool
: This WFI was trapped.

WFI_START

WFI entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFI.

WFI_WAKEUP

WFI wakeup. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: Ticks count when WFI wakeup occurred.

REASON enum
: Reason for wakeup.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.25 ARM_Cortex-A78C - trace

This section describes the trace sources.

ASYNC_MEMORY_FAULT

Context ID Register write. Fields:

FAULT unsigned int
: Fault status in ESR format

PADDR unsigned int
: Physical Address (or 0 if unavailable)

VADDR signed int
: Virtual Address (or 0 if unavailable)

ATOMIC_END_ACCESS

Bus trace access for atomics. Fields:

ACCESS_FAIL bool
: Memory access failed

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

LOAD_VALUE unsigned int
: Loaded values

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ATOMIC_START_ACCESS

Bus trace access for atomics. Fields:

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

NSDESC unsigned int
: The security state of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ArchMsg.Error.exit_code

Fields:

EXITCODE signed int
: Exit code to be returned

KIND string
: Component kind that invoked the exit code trace

REASON string
: Reason why the exit code trace was invoked (optional)

ArchMsg.Warning.IgnoredWarmResetRequest

Event fires when software attempts to request warm reset through write to deprecated DBGPRCR.CWRR.

ArchMsg.Warning.RAS_ErrselrUnpredictableState

ERRSELR is in unpredictable state due to value greater than number_of_error_records written to it.

ArchMsg.Warning.RAS_UnpredictableStateWrite

Fields:

REG_NAME string
: Name of register being written.

ArchMsg.Warning.TarmacReconstructor

Unexpected event happened during Tarmac reconstruction. Fields:

REASON string
: Reason

ArchMsg.Warning.Unpredictably Indexed PM Event Register

Fields:

IsDirect unsigned int
: Direct Access rather than indirect

N unsigned int
: Number of Accessible Registers

SEL unsigned int
: Selector

ArchMsg.Warning.branch_to_unaligned_address

Fields:

ADDR unsigned int
: Unaligned branch address

ArchMsg.Warning.cache_contents_unknown

A cache has been activated without being invalidated and may contain UNKNOWN content. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

REGIME unsigned int
: Current translation regime

SIDE unsigned int
: 0 d-side cache or unified cache, 1 i-side

ArchMsg.Warning.dcimvac_matches_watchpoint

Watchpoints matching an AArch32 DCIMVA are implementation defined.

ArchMsg.Warning.decode_fieldmismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_fracbitsnegative

VCVT Instruction is unpredictable with a negative number of fraction bits.

ArchMsg.Warning.decode_initblock

Instruction is unpredictable in an IT block.

ArchMsg.Warning.decode_initblocknotlast

Instruction is unpredictable in an IT block when not the last.

ArchMsg.Warning.decode_invalid_condition

Instruction is unpredictable when the condition is not AL.

ArchMsg.Warning.decode_invalidfieldcombination

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_invalidvalue

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_mem_hint_unallocated

There is no memory hint allocated to this bit pattern.

ArchMsg.Warning.decode_registermatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_registermismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registeroutofrange

Float point instruction is unpredictable indexing beyond the end of the register bank.

ArchMsg.Warning.decode_sbzsbo

Reserved bits in the instruction are not canonical.

ArchMsg.Warning.decode_transactiontoobig

Attempt to perform a memory transaction of more than 128 bytes.

ArchMsg.Warning.decode_unpred_other

Instruction is unpredictable.

ArchMsg.Warning.decode_unpreduseofpc

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13inreglist

Instruction is unpredictable with R13 in register-list.

ArchMsg.Warning.decode_useofd16orhigher

Attempt to use register D16 or higher where it is not allowed.

ArchMsg.Warning.decode_writebackandbaseinlist

Instruction is unpredictable with write back when the base register is in the transfer list.

ArchMsg.Warning.decode_zeroregistersinlist

Instruction is unpredictable when no registers are to be transfered.

ArchMsg.Warning.recursive_branch

An instruction is performing a branch that targets the same instruction. If this was intended then a WFI might be more effective. Fields:

PC unsigned int
: Address of instruction causing the branch

ArchMsg.Warning.recursive_exception

An instruction has generated an exception that targets the same instruction. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CPSR unsigned int
: Processor state

PC unsigned int
: Address of instruction causing the exception

TYPE enum
: Type of exception

ArchMsg.Warning.reentrant_vector_catch

Fields:

ADDR unsigned int
: vector address

ArchMsg.Warning.reserved_it_state

Fields:

ITSTATE unsigned int
: reserved IT state

ArchMsg.Warning.tlb_contents_unknown

Translation has been enabled without invalidating the TLB. Fields:

EL unsigned int
: Exception level of translation regime

INVALIDITY unsigned int
: Which TLB may be invalid (if TLBs are separate)

NON_SECURE unsigned int
: Security state of translation regime

VMID unsigned int
: Current VMID, if applicable

ArchMsg.Warning.unknown_DLR_DSPSR

Exiting debug state with unknown DLR or DSPSR.

ArchMsg.Warning.unknown_ELR_SPSR

Returning from debug exception with unknown ELR or SPSR.

ArchMsg.Warning.unpredictable_a32_breakpoint

Fields:

ADDR1 unsigned int
: Address

ADDR2 unsigned int
: Breakpoint address

BAS unsigned int
: BAS Field

IS_ADDRESS_MISMATCH_BREAKPOINT bool
: Is Mismatch breakpoint, not a match breakpoint

N unsigned int
: Breakpoint number

OFFSET unsigned int
: Indicates which halfword actually considering into the A32 instruction

ArchMsg.Warning.unpredictable_banked_register_access

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

R unsigned int
R bit
SYSM unsigned int
SYSm

ArchMsg.Warning.unpredictable_ccfail_undef

Fields:

OPCODE unsigned int
opcode of the instruction
VADDR unsigned int
Virtual address of the instruction being executed

ArchMsg.Warning.unpredictable_ici_bits

ICI bits do not apply or are architecturally unpredictable for instruction.

ArchMsg.Warning.unpredictable_t32_breakpoint

Fields:

ADDR1 unsigned int
Address
ADDR2 unsigned int
Breakpoint address
BAS unsigned int
BAS Field
IS_ADDRESS_MISMATCH_BREAKPOINT bool
Is Mismatch breakpoint, not a match breakpoint
N unsigned int
Breakpoint number
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_t32_vector_catch

Fields:

ADDR unsigned int
Address
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_watchpoint_far

Fields:

LOWER_BOUND unsigned int
the lowest address accessed by the instruction that triggered the watchpoint
UPPER_BOUND unsigned int
the highest watchpointed address accessed by that instruction

ArchMsg.Warning.warning_AdvSIMDExpandImmUnexpectedZero

AdvSIMDExpandImm may treat this immediate value as UNPREDICTABLE.

ArchMsg.Warning.warning_ConditionalSMC

SMC instruction has UNPREDICTABLE effects when conditional when combined with traps.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_access_crosses_page_boundary

Access crosses page boundary.

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault_on_both_pages

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_spanning_different_memory

Fields:

ADDR unsigned int
: address of access crossing page

MEMTYPE_PAGE1 signed int
: First Page's memory type (0-Normal, 1-Non-Normal)

MEMTYPE_PAGE2 signed int
: Second Page's memory type (0-Normal, 1-Non-Normal)

ArchMsg.Warning.warning_access_wraps_around_memory

Fields:

WIDTH unsigned int
: Address width

ArchMsg.Warning.warning_bx_from_thumbee

Fields:

ADDR unsigned int
: Destination address

ArchMsg.Warning.warning_ccsidr_unimplemented_level

Fields:

CSSELR unsigned int
: current effective value of CSSELR

ArchMsg.Warning.warning_change_to_ns_when_tge_set

Attempting to change NS to 1 when HCR.TGE = 1.

ArchMsg.Warning.warning_contiguous_bit_check_abort

Fields:

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ArchMsg.Warning.warning_contiguous_bit_error

Fields:

CONTIG_ADDR unsigned int
: address of the TLB first read, with which this entry is expected to be contiguous

CONTIG_DATA unsigned int
: expected contents based on the entry at CONTIG_ADDR

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ENTRY_DATA unsigned int
: contents of conflicting TLB entry

ArchMsg.Warning.warning_cp10_cp11_mismatch

Fields:

EL unsigned int
: EL owning the control

ArchMsg.Warning.warning_cp10_cp11_reserved_value

Fields:

CP10 unsigned int
: CP10 access

CP11 unsigned int
: CP11 access

ArchMsg.Warning.warning_csselr_level_out_of_range

Fields:

IMPLEMENTED unsigned int
: number of cache levels implemented

LEVEL unsigned int
: written value of CSSELR.Level

TYPE enum
: The type of cache selected by CSSELR.InD and TnD

ArchMsg.Warning.warning_debug_flow_control_bits_not_obeyed

Fields:

ERROR enum
: type of error resulting

REGISTER enum
: register under consideration

ArchMsg.Warning.warning_debug_register_access_during_reset

Fields:

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

ArchMsg.Warning.warning_decode_cps_inconsistent_fields

The mode setting fields in the CPS instruction are inconsistent.

ArchMsg.Warning.warning_decode_invalid_state

Instruction is unpredictable in current exception-level/security state.

ArchMsg.Warning.warning_default_cacheable_mmu_on

Default cacheable is enabled (HCR.DC = 1) while MMU is enabled.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_default_cacheable_vmmu_off

Default cacheable is enabled (HCR.DC = 1) while HCR.VM = 0.

ArchMsg.Warning.warning_deprecated_wvr_bit_2_set

Fields:

N unsigned int
: index

WVR unsigned int
: DBGWVR

ArchMsg.Warning.warning_eret_while_software_step_active_pending

Missing ISB between setting MDSCR_EL1.SS and ERET.

ArchMsg.Warning.warning_exclusive_to_non_normal

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_exclusive_to_non_writeback

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_execute_from_device_memory

An attempt was made to execute from Device memory.

ArchMsg.Warning.warning_illegal_cpsr_mode

Fields:

NEW_MODE unsigned int
: New value of CPSR.M

ArchMsg.Warning.warning_illegal_srs_mode

Fields:

MODE unsigned int
: mode for Banked SP

ArchMsg.Warning.warning_implementation_defined_read_debug_register_in_SCS

Read Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_implementation_defined_sequential_security_transitions_supported

The behaviour of a sequential instruction fetches that cross from non-secure to secure memory and contain SG instruction is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_implementation_defined_stack_limit_check_supported

It is IMPLEMENTATION DEFINED whether stack pointer limit checking is performed for the this instructions. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDRESS unsigned int
: Address of instruction

OPCODE unsigned int
: instruction opcode

ArchMsg.Warning.warning_implementation_defined_write_debug_register_in_SCS

Write Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

DATA unsigned int
: data attempted to be written

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_invalid_tcr_granule

Fields:

REQUEST signed int
: page size requested (or 0 for reserved)

SUBSTITUTE signed int
: best guess available page size

TG_ID bool
: bits TG0 or TG1

ArchMsg.Warning.warning_load_multiple_user_registers_from_user_mode

An LDM(user registers) instruction executed from user mode.

ArchMsg.Warning.warning_load_pc_from_unaligned

PC loaded from an unaligned location.

ArchMsg.Warning.warning_mte_in_mem_attr_not_supported

Fields:

ATTR unsigned int
: Memory attribute

ArchMsg.Warning.warning_reserved_breakpoint_state_match

Fields:

IS_BREAKPOINT unsigned int
: Is breakpoint, not watchpoint

ArchMsg.Warning.warning_shareability

Unpredictable: combination of the 1st and 2nd stages of translation is Normal Inner Non-Cacheable, Outer Non-Cacheable.

ArchMsg.Warning.warning_software_step_set_while_enabled

MDSCR_EL1.SS set to 1 while software step debug exceptions are enabled.

ArchMsg.Warning.warning_thumb_instruction_wraps_around_memory

Thumb instruction warps around memory.

ArchMsg.Warning.warning_ttbr_sbz_bits_are_not_zero

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MSB unsigned int
x-1
TTBR enum
which TTBR register

ArchMsg.Warning.warning_unaligned_address_dbgdtrrx_write

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_address_dbgdtrtx_read

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_device

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_strongly_ordered

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unknown_sau_rnr

SAU_RNR was set to an unsupported value.

ArchMsg.Warning.warning_unpredictable_AIRCR_PRIP_and_BFHFNMINP

The effect of setting both AIRCR.BFHFNMINP and AIRCR.PRIP to 1 is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTCLRACTIVE_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTCLRACTIVE if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_and_SYSRESETREQ

When the processor is halted in Debug state, if a write to the register writes a 1 to both VECTRESET and SYSRESETREQ, the behavior is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTRESET if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_incorrect_VKEY

The value 0x05FA must be written to AIRCR.VKEY, otherwise the register write is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_EXE_RETURN_Reserved_Bit

EXC_RETURN[23:7] are reserved with the special condition that all bits should be written as one. Values other than all 1s are UNPREDICTABLE. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

EXC_RETURN unsigned int
: Special PC value

SBO_MASK unsigned int
: Which bits Should Be One to not UNPRED

ArchMsg.Warning.warning_unpredictable_change_to_DEMCR_MON_STEP_at_insufficient_priority

The effect of changing DEMCR.MON_STEP at an execution priority that is lower than the priority of the DebugMonitor exception is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_change_to_priority_of_active_exception

Changing the priority of an active exception is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_clear_lspact

Fields:

DRVEXC_TAKEN unsigned int
: derived exception taken not pended

REACHED_LAST_STORE unsigned int
: only the last faulted

ArchMsg.Warning.warning_unpredictable_dbg_exit_unaligned_dlr

Debug exit to AArch32 with DLR[0] set to 1. PC[0] can be set to 0 or to DLR[0].

ArchMsg.Warning.warning_unpredictable_exception_catch

Generation of an exception catch event is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_inconsistent_state

state on exception return is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_instruction

exception return instruction is unpredictable.

ArchMsg.Warning.warning_unpredictable_in_debug_state

Instruction is UNPREDICTABLE when executed in debug state.

ArchMsg.Warning.warning_unpredictable_mair_encoding

Fields:

ATTR unsigned int
: Memory attribute

ArchMsg.Warning.warning_unpredictable_pmu_counter_access

Access to PMU counters from non-secure EL0 or EL1 is UNPREDICTABLE with MDCR_EL2.HPMN set to 0.

ArchMsg.Warning.warning_unpredictable_prioritization_breakpoint_match_vector_catch

Two events with the same priority occurred at the same instruction: Address Matching Vector Catch debug event, and Breakpoint debug event It is constrained unpredictable which is taken.

ArchMsg.Warning.warning_unpredictable_stack_selection

UNPREDICTABLE state on context switch.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_tsize_out_of_range

Fields:

TCR enum
: Which TCR

TSIZE signed int
: Value of TSize

TSIZE_MAX signed int
: The maximum allowed value

TTBR unsigned int
: Which TTBR region

ArchMsg.Warning.warning_unpredictable_unaligned_pc_as_base_register

The PC value must be word-aligned, otherwise the behavior of the instruction is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_unaligned_pop_stack

Fields:

FRAME_ADDR unsigned int
: unaligned stack frame address

ArchMsg.Warning.warning_unpredictable_vmsa_memattrib

Fields:

TEXCB unsigned int
: texcb value

TEX_REMAP bool
: use tex remap

ArchMsg.Warning.warning_unpredictable_vtcr_t0sz_sl0

Fields:

SL0 signed int
: VTCR.SL0, starting level for stage 2 translation table walks

T0SZ signed int
: VTCR.T0SZ, required input address range

ArchMsg.Warning.warning_unpredictable_write_DHCSR

Fields:

NEW unsigned int
: data attempted to be written

OLD unsigned int
: previous value

ArchMsg.Warning.warning_unpredictable_write_SHCSR

UNPREDICTABLE setting a pending bit to 1 for an exception with priority >= execution priority.

ArchMsg.Warning.warning_unsupported_access_to_memory_mapped_register

Access to memory mapped register is unsupported.

ArchMsg.Warning.warning_user_jmcr_access

Fields:

WRITE unsigned int
: access is write

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_wcr_mask_and_bas

DBGWCR_EL1.MASK is non-zero and BAS is not 0xff.

ArchMsg.Warning.warning_wcr_mask_reserved

DBGWCR_ELn.MASK is set to a reserved value.

ArchMsg.Warning.warning_wcr_non_configuous_bas

DBGWCR_EL1.BAS has non-contiguous set of ones.

ArchMsg.Warning.warning_wvr_masked_bit_not_zero

DBGWVRn_EL1.VA contains a bit masked by DBGWCRn_EL1.MASK that is not zero.

ArchMsg.Why.why_branch_target_exception

A branch target exception has occurred. Fields:

BTYPE unsigned int
: pstate.BTYPE at time branch taken

OPCODE unsigned int
: opcode at branch target

PC unsigned int
: Virtual Address of branch target

ArchMsg.Why.why_illegal_state

An illegal mode change has occured. Fields:

MESSAGE string
: The message

BRANCH_MISPREDICT

Simulating branch mispredict. Fields:

PC unsigned int
: Origin address (or 0 if unavailable)

BRA_DIR

Direct branches, to immediate address. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

BRA_INDIR

Indirect branches, perhaps to a register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR_CCFAIL

Indirect branches fail, perhaps to a register.

BRA_RET_CCFAIL

Branche return fail, perhaps to a register.

CACHE_MAINTENANCE_OP

Cache Maintenance Operation. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Specified MVA or Set/way

FUNCTION enum
: Clean / Invalidate

SCOPE enum
: Affected region

SIDE enum
: Inst / Data.

CCFAIL

Conditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCFAIL_UNC

Unconditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS

Conditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS_UNC

Unconditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CHECKPOINT_MESSAGE

Report error messages from the checkpointing process. Fields:

message string
: Message contents.

CHECKPOINT_RESTORE_END

Checkpoint restore completed.

CHECKPOINT_RESTORE_START

Checkpoint restore about to start.

CHECKPOINT_SAVE_END

Checkpoint save completed.

CHECKPOINT_SAVE_START

Checkpoint save about to start.

CODE_CACHE_MAINT

Code cache maintenance. Fields:

END_INCL unsigned int
: Inclusive end addres for by VA or PA requests

NS bool
: Non-secure world for PA

START unsigned int
: Start address for by VA or PA requests

TYPE enum
: Request type.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

COMPILE_BLOCK_END

Last instruction of basic block translated. Fields:

VADDR unsigned int
: Address of next instruction after this basic block

COMPILE_BLOCK_START

First instruction of basic block translated. Fields:

VADDR unsigned int
: Address of first instruction in basic block

COMPILE_INST

ARM instruction compiled. Fields:

DISASS string
: Disassembly of the instruction.

ISET enum
: The instruction set of this instruction.

ITSTATE unsigned int
: The ITSTATE current for the instruction.

OPCODE unsigned int
: The opcode of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

CONTEXTIDR

Context ID Register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

NS bool
: Secure or nonsecure banked register is accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

CONTEXT_SYNC

Called for every context synchronization event. Fields:

ARCH_CSE bool
: Architectural Context Synchronization Event

SYSREG_SYNC bool
: System register synchronization event

CORE_ENDIAN

Core BE8 Big-Endian state changed. Fields:

BE8 bool
: Core BE8 Big-Endian state

CORE_INFO

Static processor attributes. Only triggered by a call to DumpState(). Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ARCH_PROFILE enum
: The architecture profile of the core.

CLUSTER_ID unsigned int
: The cluster ID of this processor.

CORE_NUM unsigned int
: The number of this core in an MP processor.

DETAILED_CORE_FEATURES enum
: Detailed core features as a bitfield. Enum values are bitmasks to extract the info from the field

FPU_VERSION enum
: The VFP version implemented by the core.

MEM_ARCH enum
: The memory architecture of the core.

MIDR_EL1 unsigned int
: MIDR_EL1 value

NUM_CORES unsigned int
: The number of cores in this MP processor.

QUANTUM_SIZE unsigned int
: The default quantum size of the core.

SECURITY_FEATURES bool
: Does the core have security features?

CORE_LOADS

Processor load accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CORE_REGS

Changes of the core registers R0 to R14. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number, 0 to 14.

MODE enum
: Bank of the register accessed.

OLD_VALUE unsigned int
: The old value overwritten.

PHYS_ID enum
: The physical register accessed.

VALUE unsigned int
: The new value written to the register.

CORE_REGS64

Changes of the core registers X0..X30, SP_ELn. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: The register number, 0..30 for X0..X30, >=32 for SP_ELn.

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

CORE_STORES

Processor store accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CP14_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP14_WRITE

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CP15_WRITE

System Control Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

CPSR

CPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

OLD_VALUE unsigned int
: The old CPSR value

UNKNOWN unsigned int
: Bits within the register that have unknown value.

VALUE unsigned int
: The new CPSR value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CRYPTO_SPEC

Every crypto instruction speculatively executed.

CorePowerStateChange

Core power-up state has changed. Fields:

PU bool
: Whether the core powered up

DATA_CACHE_SET_ALLOC_TAGS

Set allocation tags from DC GVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of memory in bytes.

TAG unsigned int
: The value of the tag being set.

VADDR unsigned int
: Virtual address used for setting allocation tags.

DATA_CACHE_ZERO

Zero and invalidate cache line from DC ZVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of zero'd memory in bytes.

VADDR unsigned int
: Virtual address of the zero'd cache line.

DEBUG_EVENT

Hardware debug support event. Fields:

EVENT enum
: Description of event

VALUE unsigned int
: data value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

END_COMPILE

Compilation end. Fields:

END_OF_PAGE_COUNT unsigned int
: Number of basic blocks exited due to page boundary since START_COMPILE.

FETCHFAIL_COUNT unsigned int
: Number of basic blocks exited due to ifetch failure START_COMPILE.

INST_COUNT unsigned int
: Number of instructions compiled since START_COMPILE.

NONSEQ_COUNT unsigned int
: Number of basic blocks exited due to non-sequential instructions since START_COMPILE.

PAGE_STRADDLE_COUNT unsigned int
: Number of basic blocks exited due to unaligned instructions crossing page since START_COMPILE.

ESB

Error Synchronization Barrier hit. Fields:

ASYNC bool
: ESB is async

IMPLICIT bool
: ESB is implicit

EXCEPTION

Exception taken. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

IS_PHYSICAL bool
: Physical or Virtual exception

LR unsigned int
: The value assigned to the link register.

PC unsigned int
: The location where the exception occurred.

PREFERRED_RETURN unsigned int
: The preferred return address for the exception.

TARGET_ISET enum
: The instruction set of the exception handler code.

TARGET_PC unsigned int
: The address the exception branches to.

VECTOR enum
: The exception vector.

VECTOR_OFFSET unsigned int
: The offset of an exception vector from the base address.

EXCEPTION_END

Every exception completed.

EXCEPTION_RAISE

Every exception raised.

EXCEPTION_RETURN

Branches on leaving exception. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

EXCEPTION_RETURN_PREBRANCH

Exception return event that must occur before PC and processor state have updated.

EXCEPTION_START

Every exception started.

FIQ_TAKEN

FIQ taken exception.

FP_STATE

Floating point state. Fields:

FPSR unsigned int
: FPSR register value.

MASK unsigned int
: The mask indicating updated cumulative bits.

HLT

HLT instruction occurred. Fields:

IMM16 unsigned int
: The 16-bit immediate value encoded in HLT instruction.

INST

Every instruction executed. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DEBUG_STATE bool
: The instruction is executed in debug state.

DISASS string
: Disassembly of instruction.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The current instruction set.

ITSTATE unsigned int
: The current ITSTATE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

MODE enum
: The mode the core is in.

NS unsigned int
: The core's non-secure bit.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

OPCODE unsigned int
: The opcode of the instruction.

PADDR unsigned int
: The physical address of the instruction.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

INST_END

Every instruction completed.

INST_START

Every instruction started. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

INST_STRADDLE

Instruction straddles boundary. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

IRQ_TAKEN

IRQ taken exception.

ISB

ISB instruction occurred.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOAD_MULTIPLE_REGS_LIST

List of destination registers (not source) of a multiple registers load instruction. Fields:

REG string
: Register to add into the list

LOCAL_MONITOR

Local monitor activity. Fields:

PADDR unsigned int
: Local Monitor Address

State enum
: State of the monitor (Open/Exclusive)

MEMMAP_DEBUG_READ

Memory mapped reads to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

MEMMAP_DEBUG_WRITE

Memory mapped writes to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

MMU_TRANS

Address translation information. Fields:

ASID unsigned int
: Address space identifier.

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Hyp bool
: Entry matches in Hyp state only

INNERCACHE_RA bool
: Is the inner cache allocate on read

INNERCACHE_TRANSIENT bool
: Is the inner write-through transient

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

MEMTYPE enum
: Memory type.

NSDESC enum
: Is secure side supposed to access secure or nonsecure memory

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TRANSIENT bool
: Is the outer write-through transient

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

PADDR unsigned int
: Physical address of the access.

PAGESIZE unsigned int
: Page size as log2(size).

SH enum
: Shareability

SIDE enum
: Inst / Data.

STAGE1_PERM unsigned int
: Stage 1 permissions, [5:3] Privileged access (XWR) [2:0] User access (XWR)

STAGE2_PERM unsigned int
: Stage 2 permission mask [5:3] Privileged access (XWR) [2:0] User access (XWR)

VADDR unsigned int
: Virtual address of the access.

VMID unsigned int
: Virtual machine identifier.

XS bool
: XS Attribute

nG enum
: Flag indicating whether ASID will be matched

MMU_TTB_READ

This event is triggered by reads caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data read.

IPA unsigned int
: For stage 1, the IPA of the read.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

OUTADDR unsigned int
: The output physical address.

PADDR unsigned int
: The physical address of the read.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MMU_TTB_WRITE

This event is triggered by writes caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data written.

IPA unsigned int
: For stage 1, the IPA of the write.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

PADDR unsigned int
: The physical address of the write.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MODE_CHANGE

Mode change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: The new mode.

NON_SECURE enum
: The new security state

OLD_MODE enum
: The old mode.

NOT_TAG_CHECKED_INST

MTE Tag not checked for current instruction.

PERIODIC

Called for every quantum. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

PC unsigned int
: The address of the next instruction to be executed on this CPU.

PMU_COUNTER_OVERFLOW

PMU counter overflow. Fields:

EVENT_ID unsigned int
: Event Identifier

INDEX unsigned int
: Counter index (as selected by PMSELR).

INTERRUPT bool
: Is interrupt enabled on overflow for this counter

POINTER_AUTH

Called when execution hits a pointer authentication check. Fields:

AUTH_STATE bool
: Authentication result

KEY_USED enum
: Pointer authentication key

PREFETCH_MEMORY64

Prefetch from PRFM or PRFUM instructions. Fields:

PRFOP unsigned int
: Prefetch hint

VADDR unsigned int
: Virtual address of the location that should be prefetched.

PRELOAD_DATA

Data preload from PLD instruction. Fields:

VADDR unsigned int
: Virtual address of the data that should be preloaded.

PRELOAD_INST

Instruction preload from PLI instruction. Fields:

VADDR unsigned int
: Virtual address of the instruction that should be preloaded.

PRE_CORE_LOAD

Trace just before a core load. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_CORE_STORE

Trace just before a core store. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_TTB_READ

This event is triggered before reads caused by a translation table walk. Fields:

ADDR unsigned int
: The physical address of the read.

PSTATE_PACTIVE_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

PSTATE_PACTIVE_OPMODE_TRANSITION_DENIED

Trace the denied pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_POWER_TRANSITION

Trace pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

PSTATE_PACTIVE_POWER_TRANSITION_DENIED

Trace the denied pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

REASON string
: Reason for the transition being denied

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PSTATE_PACTIVE_THREAD_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions for threads on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

RUN_STATE

Run state transition. Fields:

INST_COUNT unsigned int
: Ticks count at point of transition.

NEW enum
: New run state.

OLD enum
: Old run state.

SBP_SAMPLE_COLLISION

Statistical profiling samples collided .

SBP_SAMPLE_DISCARDED

Fields:

BUFFER_POINTER unsigned int
: Value of buffer pointer

DATA unsigned int
: Data discarded

SIZE unsigned int
: Size of sample, in bytes

SBP_SAMPLE_END

Statistical profiling of an operation ended.

SBP_SAMPLE_POP

Operation which might be sampled by statistical profiling.

SBP_SAMPLE_RECORDED

Fields:

BUFFER_ADDRESS unsigned int
: Virtual Address of start of buffer write

DATA unsigned int
: Data written to the buffer

SIZE unsigned int
: Size of sample written to buffer, in bytes

SBP_SAMPLE_START

Statistical profiling of an operation started.

SBP_SAMPLE_TAKEN

Operation sampled by statistical profiling after filtering.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SEMIHOSTING_CALL

Call of a semihost function occurred. Fields:

PC unsigned int
: The program counter after the semihosting call.

SEMIHOSTING_PRECALL

About to call semihost. Fields:

EL2 bool
: Use EL2 translation regime.

NS bool
: Is Non-Secure.

REG_WIDTH unsigned int
: The current register width in bytes.

SIGNAL

External signal state change. Fields:

SIGNAL enum
: Signal that changed

STATE bool
: Signal asserted state

SOFTWARE_STEP

Return the Debug Software Step state. Fields:

STATE enum
: Software Step state

SPSR

SPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: Which of the banked SPSR registers is written.

OLD_VALUE unsigned int
: The old SPSR value

VALUE unsigned int
: The new SPSR value

START_COMPILE

Compilation started. Fields:

VADDR unsigned int
: Instruction where compilation begins.

STG_MULTIPLE_MICRO_OPS_ADDRESS_RANGE

Address range for MTE stores with multiple uops. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

STORE_MULTIPLE_REGS_ADDRESS_RANGE

Address range for stores with multiple registers. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

SUBOPTIMAL_LDST_RETIRED

suboptimal load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1: half-word, 2: word ...)

OPTIMAL_ALIGN_SELECTOR unsigned int
: Selector to decide the optimal alignment size (1: access width, 2: 4-bytes, 3: 8-bytes, ..., 12: 4-kbytes)

VADDR unsigned int
: The virtual address of the access

SYNC

Called for every synchronization, that is every quantum break. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

LOCAL_QUANTUM unsigned int
: The local quantum of this CPU.

LOCAL_TIME unsigned int
: The local time of this CPU.

SYNC_REASON enum
: Reason for sync.

SYSCALL

System call instruction executed. Fields:

IMM unsigned int
: Immediate value of the system call instruction.

TYPE enum
: System call type.

VADDR unsigned int
: Instruction that caused the system call.

SYSREG_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE32

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE64

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SYSREG_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

UNALIGNED_LDST_RETIRED

Processor unaligned load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1=>half-word, 2->word ...)

NUMBER_OF_BEATS unsigned int
: Number of accesses (beats) in this burst (Total size in bytes is ACCESS_SIZE * NUMBER_OF_BEATS)

VADDR unsigned int
: The virtual address of the access

VFP_D_REGS

VFP/NEON D 64 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VFP_Q_REGS

VFP/NEON Q 128 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding byte in the register. For example, 0x1 would indicate a write to VALUE[7:0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_SYS_REGS

Writes to the VFP/NEON units system registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: Which VFP system register is written.

MASK unsigned int
: The mask indicating updated cumulative bits.

OLD_VALUE unsigned int
: The register's old value overwritten.

VALUE unsigned int
: The new value written to the VFP system register.

VFP_S_REGS

VFP/NEON S 32 bit register write. Fields:

ALIASING enum
: Type of register aliasing used.

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

WAYPOINT

Signals end of basic block execution. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: Origin instructions set

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

IS_COND bool
: Indicates if this is a conditional waypoint.

PC unsigned int
: Origin address (or 0 if unavailable)

TAKEN bool
: Indicates if this waypoint was taken.

TARGET unsigned int
: Destination address

TARGET_ISET enum
: Destination instructions set

WFE_END

WFE ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFE.

WFE_EVENT_REGISTER

WFE event register status: set/clear, reason. Fields:

INST_COUNT unsigned int
: Ticks count.

REASON enum
: Reason for set/clear. Only REASON==0 (Cleared by WFE) clears the bit, all other reasons set it.

WFE_IGNORED

WFE ignored. Fields:

EVENT bool
: This WFE was ignored because the event register was set.

INST_COUNT unsigned int
: Ticks count when ignoring WFE.

TRAPPED bool
: This WFE was trapped.

WFE_START

WFE entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFE.

WFI_END

WFI ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFI.

WFI_IGNORED

WFI ignored. Fields:

DISABLED bool
: This WFI was ignored because WFI is disabled.

INST_COUNT unsigned int
: Ticks count when ignoring WFI.

REASON enum
: specifies reason why WFI trace was ignored

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

TRAPPED bool
: This WFI was trapped.

WFI_START

WFI entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFI.

WFI_WAKEUP

WFI wakeup. Fields:

INST_COUNT unsigned int
: Ticks count when WFI wakeup occurred.

REASON enum
: Reason for wakeup.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.26 ARM_Cortex-A8 - trace

This section describes the trace sources.

ASYNC_MEMORY_FAULT

Context ID Register write. Fields:

FAULT unsigned int
: Fault status in ESR format

PADDR unsigned int
: Physical Address (or 0 if unavailable)

VADDR signed int
: Virtual Address (or 0 if unavailable)

ATOMIC_END_ACCESS

Bus trace access for atomics. Fields:

ACCESS_FAIL bool
: Memory access failed

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

LOAD_VALUE unsigned int
: Loaded values

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ATOMIC_START_ACCESS

Bus trace access for atomics. Fields:

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

NSDESC unsigned int
: The security state of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ArchMsg.Error.exit_code

Fields:

EXITCODE signed int
: Exit code to be returned

KIND string
: Component kind that invoked the exit code trace

REASON string
: Reason why the exit code trace was invoked (optional)

ArchMsg.Warning.TarmacReconstructor

Unexpected event happened during Tarmac reconstruction. Fields:

REASON string
: Reason

ArchMsg.Warning.cache_contents_unknown

A cache has been activated without being invalidated and may contain UNKNOWN content. Fields:

REGIME unsigned int
: Current translation regime

SIDE unsigned int
: 0 d-side cache or unified cache, 1 i-side

ArchMsg.Warning.recursive_branch

An instruction is performing a branch that targets the same instruction. If this was intended then a WFI might be more effective. Fields:

PC unsigned int
: Address of instruction causing the branch

ArchMsg.Warning.recursive_exception

An instruction has generated an exception that targets the same instruction. Fields:

CPSR unsigned int
: Processor state

PC unsigned int
: Address of instruction causing the exception

TYPE enum
: Type of exception

ArchMsg.Warning.tlb_contents_unknown

Translation has been enabled without invalidating the TLB. Fields:

EL unsigned int
: Exception level of translation regime

INVALIDITY unsigned int
: Which TLB may be invalid (if TLBs are separate)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NON_SECURE unsigned int
: Security state of translation regime

VMID unsigned int
: Current VMID, if applicable

BRANCH_MISPREDICT

Simulating branch mispredict. Fields:

PC unsigned int
: Origin address (or 0 if unavailable)

BRA_DIR

Direct branches, to immediate address. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR

Indirect branches, perhaps to a register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR_CCFAIL

Indirect branches fail, perhaps to a register.

BRA_RET_CCFAIL

Branche return fail, perhaps to a register.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CACHE_MAINTENANCE_OP

Cache Maintenance Operation. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Specified MVA or Set/way

FUNCTION enum
: Clean / Invalidate

SCOPE enum
: Affected region

SIDE enum
: Inst / Data.

CCFAIL

Conditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCFAIL_UNC

Unconditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS

Conditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS_UNC

Unconditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CODE_CACHE_MAINT

Code cache maintenance. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

END_INCL unsigned int
: Inclusive end addres for by VA or PA requests

NS bool
: Non-secure world for PA

START unsigned int
: Start address for by VA or PA requests

TYPE enum
: Request type.

COMPILE_BLOCK_END

Last instruction of basic block translated. Fields:

VADDR unsigned int
: Address of next instruction after this basic block

COMPILE_BLOCK_START

First instruction of basic block translated. Fields:

VADDR unsigned int
: Address of first instruction in basic block

COMPILE_INST

ARM instruction compiled. Fields:

DISASS string
: Disassembly of the instruction.

ISET enum
: The instruction set of this instruction.

ITSTATE unsigned int
: The ITSTATE current for the instruction.

OPCODE unsigned int
: The opcode of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

CONTEXTIDR

Context ID Register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

NS bool
: Secure or nonsecure banked register is accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

CONTEXT_SYNC

Called for every context synchronization event. Fields:

ARCH_CSE bool
: Architectural Context Synchronization Event

SYSREG_SYNC bool
: System register synchronization event

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_ENDIAN

Core BE8 Big-Endian state changed. Fields:

BE8 bool
: Core BE8 Big-Endian state

CORE_INFO

Static processor attributes. Only triggered by a call to DumpState(). Fields:

ARCH_PROFILE enum
: The architecture profile of the core.

CLUSTER_ID unsigned int
: The cluster ID of this processor.

CORE_NUM unsigned int
: The number of this core in an MP processor.

DETAILED_CORE_FEATURES enum
: Detailed core features as a bitfield. Enum values are bitmasks to extract the info from the field

FPU_VERSION enum
: The VFP version implemented by the core.

MEM_ARCH enum
: The memory architecture of the core.

MIDR_EL1 unsigned int
: MIDR_EL1 value

NUM_CORES unsigned int
: The number of cores in this MP processor.

QUANTUM_SIZE unsigned int
: The default quantum size of the core.

SECURITY_FEATURES bool
: Does the core have security features?

CORE_LOADS

Processor load accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CORE_REGS

Changes of the core registers R0 to R14. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number, 0 to 14.

MODE enum
: Bank of the register accessed.

OLD_VALUE unsigned int
: The old value overwritten.

PHYS_ID enum
: The physical register accessed.

VALUE unsigned int
: The new value written to the register.

CORE_STORES

Processor store accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CP14_READ

Reads to the System Coprocessor registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP14_WRITE

Writes to the System Coprocessor registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

CP15_WRITE

System Control Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

CPSR

CPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

OLD_VALUE unsigned int
: The old CPSR value

UNKNOWN unsigned int
: Bits within the register that have unknown value.

VALUE unsigned int
: The new CPSR value

CRYPTO_SPEC

Every crypto instruction speculatively executed.

DATA_CACHE_SET_ALLOC_TAGS

Set allocation tags from DC GVA instruction. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of memory in bytes.

TAG unsigned int
: The value of the tag being set.

VADDR unsigned int
: Virtual address used for setting allocation tags.

DATA_CACHE_ZERO

Zero and invalidate cache line from DC ZVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of zero'd memory in bytes.

VADDR unsigned int
: Virtual address of the zero'd cache line.

END_COMPILE

Compilation end. Fields:

END_OF_PAGE_COUNT unsigned int
: Number of basic blocks exited due to page boundary since START_COMPILE.

FETCHFAIL_COUNT unsigned int
: Number of basic blocks exited due to ifetch failure START_COMPILE.

INST_COUNT unsigned int
: Number of instructions compiled since START_COMPILE.

NONSEQ_COUNT unsigned int
: Number of basic blocks exited due to non-sequential instructions since START_COMPILE.

PAGE_STRADDLE_COUNT unsigned int
: Number of basic blocks exited due to unaligned instructions crossing page since START_COMPILE.

EXCEPTION

Exception taken. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

IS_PHYSICAL bool
: Physical or Virtual exception

LR unsigned int
: The value assigned to the link register.

PC unsigned int
: The location where the exception occurred.

PREFERRED_RETURN unsigned int
: The preferred return address for the exception.

TARGET_ISET enum
: The instruction set of the exception handler code.

TARGET_PC unsigned int
: The address the exception branches to.

VECTOR enum
: The exception vector.

VECTOR_OFFSET unsigned int
: The offset of an exception vector from the base address.

EXCEPTION_END

Every exception completed.

EXCEPTION_RAISE

Every exception raised.

EXCEPTION_RETURN

Branches on leaving exception. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

EXCEPTION_RETURN_PREBRANCH

Exception return event that must occur before PC and processor state have updated.

EXCEPTION_START

Every exception started.

FP_STATE

Floating point state. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

FPSR unsigned int
: FPSR register value.

MASK unsigned int
: The mask indicating updated cumulative bits.

HLT

HLT instruction occurred. Fields:

IMM16 unsigned int
: The 16-bit immediate value encoded in HLT instruction.

INST

Every instruction executed. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DEBUG_STATE bool
: The instruction is executed in debug state.

DISASS string
: Disassembly of instruction.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The current instruction set.

ITSTATE unsigned int
: The current ITSTATE.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

MODE enum
: The mode the core is in.

NS unsigned int
: The core's non-secure bit.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

OPCODE unsigned int
: The opcode of the instruction.

PADDR unsigned int
: The physical address of the instruction.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

INST_END

Every instruction completed.

INST_START

Every instruction started. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

INST_STRADDLE

Instruction straddles boundary. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

ISB

ISB instruction occurred.

LOAD_MULTIPLE_REGS_LIST

List of destination registers (not source) of a multiple registers load instruction. Fields:

REG string
: Register to add into the list

LOCAL_MONITOR

Local monitor activity. Fields:

PADDR unsigned int
: Local Monitor Address

State enum
: State of the monitor (Open/Exclusive)

MMU_TRANS

Address translation information. Fields:

ASID unsigned int
: Address space identifier.

CORE_NUM unsigned int
: Core number in a multi processor.

Hyp bool
: Entry matches in Hyp state only

INNERCACHE_RA bool
: Is the inner cache allocate on read

INNERCACHE_TRANSIENT bool
: Is the inner write-through transient

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MEMTYPE enum
: Memory type.

NSDESC enum
: Is secure side supposed to access secure or nonsecure memory

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TRANSIENT bool
: Is the outer write-through transient

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

PADDR unsigned int
: Physical address of the access.

PAGESIZE unsigned int
: Page size as log2(size).

SH enum
: Shareability

SIDE enum
: Inst / Data.

STAGE1_PERM unsigned int
: Stage 1 permissions, [5:3] Privileged access (XWR) [2:0] User access (XWR)

STAGE2_PERM unsigned int
: Stage 2 permission mask [5:3] Privileged access (XWR) [2:0] User access (XWR)

VADDR unsigned int
: Virtual address of the access.

VMID unsigned int
: Virtual machine identifier.

XS bool
: XS Attribute

nG enum
: Flag indicating whether ASID will be matched

MMU_TTB_READ

This event is triggered by reads caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data read.

IPA unsigned int
: For stage 1, the IPA of the read.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

OUTADDR unsigned int
: The output physical address.

PADDR unsigned int
: The physical address of the read.

SIDE enum
: Inst / Data.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

STAGE unsigned int
: Translation stage

MMU_TTB_WRITE

This event is triggered by writes caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data written.

IPA unsigned int
: For stage 1, the IPA of the write.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

PADDR unsigned int
: The physical address of the write.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MODE_CHANGE

Mode change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: The new mode.

NON_SECURE enum
: The new security state

OLD_MODE enum
: The old mode.

NOT_TAG_CHECKED_INST

MTE Tag not checked for current instruction.

PERIODIC

Called for every quantum. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

PC unsigned int
: The address of the next instruction to be executed on this CPU.

PRELOAD_DATA

Data preload from PLD instruction. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VADDR unsigned int
: Virtual address of the data that should be preloaded.

PRELOAD_INST

Instruction preload from PLI instruction. Fields:

VADDR unsigned int
: Virtual address of the instruction that should be preloaded.

PRE_CORE_LOAD

Trace just before a core load. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_CORE_STORE

Trace just before a core store. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_TTB_READ

This event is triggered before reads caused by a translation table walk. Fields:

ADDR unsigned int
: The physical address of the read.

PSTATE_PACTIVE_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

PSTATE_PACTIVE_OPMODE_TRANSITION_DENIED

Trace the denied pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_POWER_TRANSITION

Trace pactive and power mode transitions on PChannel. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

PSTATE_PACTIVE_POWER_TRANSITION_DENIED

Trace the denied pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_THREAD_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions for threads on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

RUN_STATE

Run state transition. Fields:

INST_COUNT unsigned int
: Ticks count at point of transition.

NEW enum
: New run state.

OLD enum
: Old run state.

SEMIHOSTING_CALL

Call of a semihost function occurred. Fields:

PC unsigned int
: The program counter after the semihosting call.

SEMIHOSTING_PRECALL

About to call semihost. Fields:

EL2 bool
: Use EL2 translation regime.

NS bool
: Is Non-Secure.

REG_WIDTH unsigned int
: The current register width in bytes.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SIGNAL

External signal state change. Fields:

SIGNAL enum
: Signal that changed

STATE bool
: Signal asserted state

SPSR

SPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: Which of the banked SPSR registers is written.

OLD_VALUE unsigned int
: The old SPSR value

VALUE unsigned int
: The new SPSR value

START_COMPILE

Compilation started. Fields:

VADDR unsigned int
: Instruction where compilation begins.

STG_MULTIPLE_MICRO_OPS_ADDRESS_RANGE

Address range for MTE stores with multiple uops. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

STORE_MULTIPLE_REGS_ADDRESS_RANGE

Address range for stores with multiple registers. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

SUBOPTIMAL_LDST_RETIRED

suboptimal load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1: half-word, 2: word ...)

OPTIMAL_ALIGN_SELECTOR unsigned int
: Selector to decide the optimal alignment size (1: access width, 2: 4-bytes, 3: 8-bytes, ..., 12: 4-kbytes)

VADDR unsigned int
: The virtual address of the access

SYNC

Called for every synchronization, that is every quantum break. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

LOCAL_QUANTUM unsigned int
: The local quantum of this CPU.

LOCAL_TIME unsigned int
: The local time of this CPU.

SYNC_REASON enum
: Reason for sync.

SYSCALL

System call instruction executed. Fields:

IMM unsigned int
: Immediate value of the system call instruction.

TYPE enum
: System call type.

VADDR unsigned int
: Instruction that caused the system call.

SYSREG_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE32

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE64

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

UNALIGNED_LDST_RETIRED

Processor unaligned load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1=>half-word, 2->word ...)

NUMBER_OF_BEATS unsigned int
: Number of accesses (beats) in this burst (Total size in bytes is ACCESS_SIZE * NUMBER_OF_BEATS)

VADDR unsigned int
: The virtual address of the access

VFP_D_REGS

VFP/NEON D 64 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_Q_REGS

VFP/NEON Q 128 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding byte in the register. For example, 0x1 would indicate a write to VALUE[7:0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_SYS_REGS

Writes to the VFP/NEON units system registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ID enum
: Which VFP system register is written.

MASK unsigned int
: The mask indicating updated cumulative bits.

OLD_VALUE unsigned int
: The register's old value overwritten.

VALUE unsigned int
: The new value written to the VFP system register.

VFP_S_REGS

VFP/NEON S 32 bit register write. Fields:

ALIASING enum
: Type of register aliasing used.

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

WAYPOINT

Signals end of basic block execution. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: Origin instructions set

IS_COND bool
: Indicates if this is a conditional waypoint.

PC unsigned int
: Origin address (or 0 if unavailable)

TAKEN bool
: Indicates if this waypoint was taken.

TARGET unsigned int
: Destination address

TARGET_ISET enum
: Destination instructions set

WFE_END

WFE ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFE.

WFE_EVENT_REGISTER

WFE event register status: set/clear, reason. Fields:

INST_COUNT unsigned int
: Ticks count.

REASON enum
: Reason for set/clear. Only REASON==0 (Cleared by WFE) clears the bit, all other reasons set it.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

WFE_IGNORED

WFE ignored. Fields:

EVENT bool
: This WFE was ignored because the event register was set.

INST_COUNT unsigned int
: Ticks count when ignoring WFE.

TRAPPED bool
: This WFE was trapped.

WFE_START

WFE entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFE.

WFI_END

WFI ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFI.

WFI_IGNORED

WFI ignored. Fields:

DISABLED bool
: This WFI was ignored because WFI is disabled.

INST_COUNT unsigned int
: Ticks count when ignoring WFI.

REASON enum
: specifies reason why WFI trace was ignored

TRAPPED bool
: This WFI was trapped.

WFI_START

WFI entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFI.

WFI_WAKEUP

WFI wakeup. Fields:

INST_COUNT unsigned int
: Ticks count when WFI wakeup occurred.

REASON enum
: Reason for wakeup.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.27 ARM_Cortex-A9MP - trace

This section describes the trace sources.

ASYNC_MEMORY_FAULT

Context ID Register write. Fields:

FAULT unsigned int
: Fault status in ESR format

PADDR unsigned int
: Physical Address (or 0 if unavailable)

VADDR signed int
: Virtual Address (or 0 if unavailable)

ATOMIC_END_ACCESS

Bus trace access for atomics. Fields:

ACCESS_FAIL bool
: Memory access failed

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

LOAD_VALUE unsigned int
: Loaded values

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ATOMIC_START_ACCESS

Bus trace access for atomics. Fields:

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

NSDESC unsigned int
: The security state of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ArchMsg.Error.exit_code

Fields:

EXITCODE signed int
: Exit code to be returned

KIND string
: Component kind that invoked the exit code trace

REASON string
: Reason why the exit code trace was invoked (optional)

ArchMsg.Warning.IgnoredWarmResetRequest

Event fires when software attempts to request warm reset through write to deprecated DBGPRCR.CWRR.

ArchMsg.Warning.TarmacReconstructor

Unexpected event happened during Tarmac reconstruction. Fields:

REASON string
: Reason

ArchMsg.Warning.Unpredictably Indexed PM Event Register

Fields:

IsDirect unsigned int
: Direct Access rather than indirect

N unsigned int
: Number of Accessible Registers

SEL unsigned int
: Selector

ArchMsg.Warning.branch_to_unaligned_address

Fields:

ADDR unsigned int
: Unaligned branch address

ArchMsg.Warning.cache_contents_unknown

A cache has been activated without being invalidated and may contain UNKNOWN content. Fields:

REGIME unsigned int
: Current translation regime

SIDE unsigned int
: 0 d-side cache or unified cache, 1 i-side

ArchMsg.Warning.dcimvac_matches_watchpoint

Watchpoints matching an AArch32 DCIMVA are implementation defined.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_fieldmismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_fracbitsnegative

VCVT Instruction is unpredictable with a negative number of fraction bits.

ArchMsg.Warning.decode_initblock

Instruction is unpredictable in an IT block.

ArchMsg.Warning.decode_initblocknotlast

Instruction is unpredictable in an IT block when not the last.

ArchMsg.Warning.decode_invalid_condition

Instruction is unpredictable when the condition is not AL.

ArchMsg.Warning.decode_invalidfieldcombination

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_invalidvalue

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_mem_hint_unallocated

There is no memory hint allocated to this bit pattern.

ArchMsg.Warning.decode_registermatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registermismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registeroutofrange

Float point instruction is unpredictable indexing beyond the end of the register bank.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_sbzsbo

Reserved bits in the instruction are not canonical.

ArchMsg.Warning.decode_transactiontoobig

Attempt to perform a memory transaction of more than 128 bytes.

ArchMsg.Warning.decode_unpred_other

Instruction is unpredictable.

ArchMsg.Warning.decode_unpreduseofpc

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13inreglist

Instruction is unpredictable with R13 in register-list.

ArchMsg.Warning.decode_useofd16orhigher

Attempt to use register D16 or higher where it is not allowed.

ArchMsg.Warning.decode_writebackandbaseinlist

Instruction is unpredictable with write back when the base register is in the transfer list.

ArchMsg.Warning.decode_zeroregistersinlist

Instruction is unpredictable when no registers are to be transfered.

ArchMsg.Warning.recursive_branch

An instruction is performing a branch that targets the same instruction. If this was intended then a WFI might be more effective. Fields:

PC unsigned int
: Address of instruction causing the branch

ArchMsg.Warning.recursive_exception

An instruction has generated an exception that targets the same instruction. Fields:

CPSR unsigned int
: Processor state

PC unsigned int
: Address of instruction causing the exception

TYPE enum
: Type of exception

ArchMsg.Warning.reentrant_vector_catch

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR unsigned int
vector address

ArchMsg.Warning.reserved_it_state

Fields:

ITSTATE unsigned int
reserved IT state

ArchMsg.Warning.tlb_contents_unknown

Translation has been enabled without invalidating the TLB. Fields:

EL unsigned int
Exception level of translation regime
INVALIDITY unsigned int
Which TLB may be invalid (if TLBs are separate)
NON_SECURE unsigned int
Security state of translation regime
VMID unsigned int
Current VMID, if applicable

ArchMsg.Warning.unpredictable_a32_breakpoint

Fields:

ADDR1 unsigned int
Address
ADDR2 unsigned int
Breakpoint address
BAS unsigned int
BAS Field
IS_ADDRESS_MISMATCH_BREAKPOINT bool
Is Mismatch breakpoint, not a match breakpoint
N unsigned int
Breakpoint number
OFFSET unsigned int
Indicates which halfword actually considering into the A32 instruction

ArchMsg.Warning.unpredictable_banked_register_access

Fields:

R unsigned int
R bit
SYSM unsigned int
SYSm

ArchMsg.Warning.unpredictable_ccfail_undef

Fields:

OPCODE unsigned int
opcode of the instruction
VADDR unsigned int
Virtual address of the instruction being executed

ArchMsg.Warning.unpredictable_ici_bits

ICI bits do not apply or are architecturally unpredictable for instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.unpredictable_t32_breakpoint

Fields:

ADDR1 unsigned int
: Address

ADDR2 unsigned int
: Breakpoint address

BAS unsigned int
: BAS Field

IS_ADDRESS_MISMATCH_BREAKPOINT bool
: Is Mismatch breakpoint, not a match breakpoint

N unsigned int
: Breakpoint number

OFFSET unsigned int
: Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_t32_vector_catch

Fields:

ADDR unsigned int
: Address

OFFSET unsigned int
: Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_watchpoint_far

Fields:

LOWER_BOUND unsigned int
: the lowest address accessed by the instruction that triggered the watchpoint

UPPER_BOUND unsigned int
: the highest watchpointed address accessed by that instruction

ArchMsg.Warning.warning_AdvSIMDExpandImmUnexpectedZero

AdvSIMDExpandImm may treat this immediate value as UNPREDICTABLE.

ArchMsg.Warning.warning_ConditionalSMC

SMC instruction has UNPREDICTABLE effects when conditional when combined with traps.

ArchMsg.Warning.warning_access_crosses_page_boundary

Access crosses page boundary.

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault_on_both_pages

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_spanning_different_memory

Fields:

ADDR unsigned int
: address of access crossing page

MEMTYPE_PAGE1 signed int
: First Page's memory type (0-Normal, 1-Non-Normal)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MEMTYPE_PAGE2 signed int
Second Page's memory type (0-Normal, 1-Non-Normal)

ArchMsg.Warning.warning_access_wraps_around_memory

Fields:

WIDTH unsigned int
Address width

ArchMsg.Warning.warning_bx_from_thumbee

Fields:

ADDR unsigned int
Destination address

ArchMsg.Warning.warning_ccsidr_unimplemented_level

Fields:

CSSELR unsigned int
current effective value of CSSELR

ArchMsg.Warning.warning_change_to_ns_when_tge_set

Attempting to change NS to 1 when HCR.TGE = 1.

ArchMsg.Warning.warning_contiguous_bit_check_abort

Fields:

ENTRY_ADDR unsigned int
address of conflicting TLB entry

ArchMsg.Warning.warning_contiguous_bit_error

Fields:

CONTIG_ADDR unsigned int
address of the TLB first read, with which this entry is expected to be contiguous
CONTIG_DATA unsigned int
expected contents based on the entry at CONTIG_ADDR
ENTRY_ADDR unsigned int
address of conflicting TLB entry
ENTRY_DATA unsigned int
contents of conflicting TLB entry

ArchMsg.Warning.warning_cp10_cp11_mismatch

Fields:

EL unsigned int
EL owning the control

ArchMsg.Warning.warning_cp10_cp11_reserved_value

Fields:

CP10 unsigned int
CP10 access
CP11 unsigned int
CP11 access

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_csselr_level_out_of_range

Fields:

IMPLEMENTED unsigned int
: number of cache levels implemented

LEVEL unsigned int
: written value of CSSELR.Level

TYPE enum
: The type of cache selected by CSSELR.InD and TnD

ArchMsg.Warning.warning_debug_flow_control_bits_not_obeyed

Fields:

ERROR enum
: type of error resulting

REGISTER enum
: register under consideration

ArchMsg.Warning.warning_debug_register_access_during_reset

Fields:

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

ArchMsg.Warning.warning_decode_cps_inconsistent_fields

The mode setting fields in the CPS instruction are inconsistent.

ArchMsg.Warning.warning_decode_invalid_state

Instruction is unpredictable in current exception-level/security state.

ArchMsg.Warning.warning_default_cacheable_mmu_on

Default cacheable is enabled (HCR.DC = 1) while MMU is enabled.

ArchMsg.Warning.warning_default_cacheable_vmmu_off

Default cacheable is enabled (HCR.DC = 1) while HCR.VM = 0.

ArchMsg.Warning.warning_deprecated_wvr_bit_2_set

Fields:

N unsigned int
: index

WVR unsigned int
: DBGWVR

ArchMsg.Warning.warning_eret_while_software_step_active_pending

Missing ISB between setting MDSCR_EL1.SS and ERET.

ArchMsg.Warning.warning_exclusive_to_non_normal

Fields:

ADDR unsigned int
: Address

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_exclusive_to_non_writeback

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_execute_from_device_memory

An attempt was made to execute from Device memory.

ArchMsg.Warning.warning_illegal_cpsr_mode

Fields:

NEW_MODE unsigned int
: New value of CPSR.M

ArchMsg.Warning.warning_illegal_srs_mode

Fields:

MODE unsigned int
: mode for Banked SP

ArchMsg.Warning.warning_implementation_defined_read_debug_register_in_SCS

Read Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_implementation_defined_sequential_security_transitions_supported

The behaviour of a sequential instruction fetches that cross from non-secure to secure memory and contain SG instruction is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_implementation_defined_stack_limit_check_supported

It is IMPLEMENTATION DEFINED whether stack pointer limit checking is performed for the this instructions. Fields:

ADDRESS unsigned int
: Address of instruction

OPCODE unsigned int
: instruction opcode

ArchMsg.Warning.warning_implementation_defined_write_debug_register_in_SCS

Write Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

DATA unsigned int
: data attempted to be written

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_invalid_tcr_granule

Fields:

REQUEST signed int
: page size requested (or 0 for reserved)

SUBSTITUTE signed int
: best guess available page size

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

TG_ID bool
bits TG0 or TG1

ArchMsg.Warning.warning_load_multiple_user_registers_from_user_mode

An LDM(user registers) instruction executed from user mode.

ArchMsg.Warning.warning_load_pc_from_unaligned

PC loaded from an unaligned location.

ArchMsg.Warning.warning_mte_in_mem_attr_not_supported

Fields:

ATTR unsigned int
Memory attribute

ArchMsg.Warning.warning_reserved_breakpoint_state_match

Fields:

IS_BREAKPOINT unsigned int
Is breakpoint, not watchpoint

ArchMsg.Warning.warning_shareability

Unpredictable: combination of the 1st and 2nd stages of translation is Normal Inner Non-Cacheable, Outer Non-Cacheable.

ArchMsg.Warning.warning_software_step_set_while_enabled

MDSCR_EL1.SS set to 1 while software step debug exceptions are enabled.

ArchMsg.Warning.warning_thumb_instruction_wraps_around_memory

Thumb instruction warps around memory.

ArchMsg.Warning.warning_ttbr_sbz_bits_are_not_zero

Fields:

MSB unsigned int
x-1
TTBR enum
which TTBR register

ArchMsg.Warning.warning_unaligned_address_dbgdtrrx_write

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_address_dbgdtrtx_read

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_device

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_strongly_ordered

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unknown_sau_rnr

SAU_RNR was set to an unsupported value.

ArchMsg.Warning.warning_unpredictable_AIRCR_PRIP_and_BFHFNMINP

The effect of setting both AIRCR.BFHFNMINP and AIRCR.PRIP to 1 is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTCLRACTIVE_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTCLRACTIVE if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_and_SYSRESETREQ

When the processor is halted in Debug state, if a write to the register writes a 1 to both VECTRESET and SYSRESETREQ, the behavior is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTRESET if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_incorrect_VKEY

The value 0x05FA must be written to AIRCR.VKEY, otherwise the register write is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_EXE_RETURN_Reserved_Bit

EXC_RETURN[23:7] are reserved with the special condition that all bits should be written as one. Values other than all 1s are UNPREDICTABLE. Fields:

EXC_RETURN unsigned int
Special PC value
SBO_MASK unsigned int
Which bits Should Be One to not UNPRED

ArchMsg.Warning.warning_unpredictable_change_to_DEMCR_MON_STEP_at_insufficient_priority

The effect of changing DEMCR.MON_STEP at an execution priority that is lower than the priority of the DebugMonitor exception is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_change_to_priority_of_active_exception

Changing the priority of an active exception is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_clear_lspact

Fields:

DRVEXC_TAKEN unsigned int
derived exception taken not pended

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

REACHED_LAST_STORE unsigned int
only the last faulted

ArchMsg.Warning.warning_unpredictable_dbg_exit_unaligned_dlr

Debug exit to AArch32 with DLR[0] set to 1. PC[0] can be set to 0 or to DLR[0].

ArchMsg.Warning.warning_unpredictable_exception_catch

Generation of an exception catch event is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_inconsistent_state

state on exception return is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_instruction

exception return instruction is unpredictable.

ArchMsg.Warning.warning_unpredictable_in_debug_state

Instruction is UNPREDICTABLE when executed in debug state.

ArchMsg.Warning.warning_unpredictable_mair_encoding

Fields:

ATTR unsigned int
Memory attribute

ArchMsg.Warning.warning_unpredictable_pmu_counter_access

Access to PMU counters from non-secure EL0 or EL1 is UNPREDICTABLE with MDCR_EL2.HPMN set to 0.

ArchMsg.Warning.warning_unpredictable_prioritization_breakpoint_match_vector_catch

Two events with the same priority occurred at the same instruction: Address Matching Vector Catch debug event, and Breakpoint debug event It is constrained unpredictable which is taken.

ArchMsg.Warning.warning_unpredictable_stack_selection

UNPREDICTABLE state on context switch.

ArchMsg.Warning.warning_unpredictable_tsize_out_of_range

Fields:

TCR enum
Which TCR
TSIZE signed int
Value of TSize
TSIZE_MAX signed int
The maximum allowed value
TTBR unsigned int
Which TTBR region

ArchMsg.Warning.warning_unpredictable_unaligned_pc_as_base_register

The PC value must be word-aligned, otherwise the behavior of the instruction is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_unaligned_pop_stack

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

FRAME_ADDR unsigned int
: unaligned stack frame address

ArchMsg.Warning.warning_unpredictable_vmsa_memattrib

Fields:

TEXCB unsigned int
: texcb value

TEX_REMAP bool
: use tex remap

ArchMsg.Warning.warning_unpredictable_vtcr_t0sz_sl0

Fields:

SL0 signed int
: VTCR.SL0, starting level for stage 2 translation table walks

T0SZ signed int
: VTCR.T0SZ, required input address range

ArchMsg.Warning.warning_unpredictable_write_DHCSR

Fields:

NEW unsigned int
: data attempted to be written

OLD unsigned int
: previous value

ArchMsg.Warning.warning_unpredictable_write_SHCSR

UNPREDICTABLE setting a pending bit to 1 for an exception with priority >= execution priority.

ArchMsg.Warning.warning_unsupported_access_to_memory_mapped_register

Access to memory mapped register is unsupported.

ArchMsg.Warning.warning_user_jmcr_access

Fields:

WRITE unsigned int
: access is write

ArchMsg.Warning.warning_wcr_mask_and_bas

DBGWCR_EL1.MASK is non-zero and BAS is not 0xff.

ArchMsg.Warning.warning_wcr_mask_reserved

DBGWCR_ELn.MASK is set to a reserved value.

ArchMsg.Warning.warning_wcr_non_configuous_bas

DBGWCR_EL1.BAS has non-contiguous set of ones.

ArchMsg.Warning.warning_wvr_masked_bit_not_zero

DBGWVRn_EL1.VA contains a bit masked by DBGWCRn_EL1.MASK that is not zero.

BRANCH_MISPREDICT

Simulating branch mispredict. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PC unsigned int
: Origin address (or 0 if unavailable)

BRA_DIR

Direct branches, to immediate address. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR

Indirect branches, perhaps to a register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR_CCFAIL

Indirect branches fail, perhaps to a register.

BRA_RET_CCFAIL

Branche return fail, perhaps to a register.

CACHE_MAINTENANCE_OP

Cache Maintenance Operation. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Specified MVA or Set/way

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

FUNCTION enum
: Clean / Invalidate

SCOPE enum
: Affected region

SIDE enum
: Inst / Data.

CCFAIL

Conditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCFAIL_UNC

Unconditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS

Conditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS_UNC

Unconditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CODE_CACHE_MAINT

Code cache maintenance. Fields:

END_INCL unsigned int
: Inclusive end addres for by VA or PA requests

NS bool
: Non-secure world for PA

START unsigned int
: Start address for by VA or PA requests

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

TYPE enum
Request type.

COMPILE_BLOCK_END

Last instruction of basic block translated. Fields:

VADDR unsigned int
Address of next instruction after this basic block

COMPILE_BLOCK_START

First instruction of basic block translated. Fields:

VADDR unsigned int
Address of first instruction in basic block

COMPILE_INST

ARM instruction compiled. Fields:

DISASS string
Disassembly of the instruction.
ISET enum
The instruction set of this instruction.
ITSTATE unsigned int
The ITSTATE current for the instruction.
OPCODE unsigned int
The opcode of the instruction.
PC unsigned int
The address of the instruction.
SIZE unsigned int
The size of the instruction in bytes.

CONTEXTIDR

Context ID Register write. Fields:

CORE_NUM unsigned int
Core number in a multi processor.
NS bool
Secure or nonsecure banked register is accessed.
UNDEF bool
The register accessed is undefined.
VALUE unsigned int
The new value written.

CONTEXT_SYNC

Called for every context synchronization event. Fields:

ARCH_CSE bool
Architectural Context Synchronization Event
SYSREG_SYNC bool
System register synchronization event

CORE_ENDIAN

Core BE8 Big-Endian state changed. Fields:

BE8 bool
Core BE8 Big-Endian state

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_INFO

Static processor attributes. Only triggered by a call to DumpState(). Fields:

ARCH_PROFILE enum
: The architecture profile of the core.

CLUSTER_ID unsigned int
: The cluster ID of this processor.

CORE_NUM unsigned int
: The number of this core in an MP processor.

DETAILED_CORE_FEATURES enum
: Detailed core features as a bitfield. Enum values are bitmasks to extract the info from the field

FPU_VERSION enum
: The VFP version implemented by the core.

MEM_ARCH enum
: The memory architecture of the core.

MIDR_EL1 unsigned int
: MIDR_EL1 value

NUM_CORES unsigned int
: The number of cores in this MP processor.

QUANTUM_SIZE unsigned int
: The default quantum size of the core.

SECURITY_FEATURES bool
: Does the core have security features?

CORE_LOADS

Processor load accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CORE_REGS

Changes of the core registers R0 to R14. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number, 0 to 14.

MODE enum
: Bank of the register accessed.

OLD_VALUE unsigned int
: The old value overwritten.

PHYS_ID enum
: The physical register accessed.

VALUE unsigned int
: The new value written to the register.

CORE_STORES

Processor store accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CP14_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP14_WRITE

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

CP15_WRITE

System Control Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

CPSR

CPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

OLD_VALUE unsigned int
: The old CPSR value

UNKNOWN unsigned int
: Bits within the register that have unknown value.

VALUE unsigned int
: The new CPSR value

CRYPTO_SPEC

Every crypto instruction speculatively executed.

CorePowerStateChange

Core power-up state has changed. Fields:

PU bool
: Whether the core powered up

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DATA_CACHE_SET_ALLOC_TAGS

Set allocation tags from DC GVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of memory in bytes.

TAG unsigned int
: The value of the tag being set.

VADDR unsigned int
: Virtual address used for setting allocation tags.

DATA_CACHE_ZERO

Zero and invalidate cache line from DC ZVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of zero'd memory in bytes.

VADDR unsigned int
: Virtual address of the zero'd cache line.

DEBUG_EVENT

Hardware debug support event. Fields:

EVENT enum
: Description of event

VALUE unsigned int
: data value

END_COMPILE

Compilation end. Fields:

END_OF_PAGE_COUNT unsigned int
: Number of basic blocks exited due to page boundary since START_COMPILE.

FETCHFAIL_COUNT unsigned int
: Number of basic blocks exited due to ifetch failure START_COMPILE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: Number of instructions compiled since START_COMPILE.

NONSEQ_COUNT unsigned int
: Number of basic blocks exited due to non-sequential instructions since START_COMPILE.

PAGE_STRADDLE_COUNT unsigned int
: Number of basic blocks exited due to unaligned instructions crossing page since START_COMPILE.

EXCEPTION

Exception taken. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

IS_PHYSICAL bool
: Physical or Virtual exception

LR unsigned int
: The value assigned to the link register.

PC unsigned int
: The location where the exception occurred.

PREFERRED_RETURN unsigned int
: The preferred return address for the exception.

TARGET_ISET enum
: The instruction set of the exception handler code.

TARGET_PC unsigned int
: The address the exception branches to.

VECTOR enum
: The exception vector.

VECTOR_OFFSET unsigned int
: The offset of an exception vector from the base address.

EXCEPTION_END

Every exception completed.

EXCEPTION_RAISE

Every exception raised.

EXCEPTION_RETURN

Branches on leaving exception. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

EXCEPTION_RETURN_PREBRANCH

Exception return event that must occur before PC and processor state have updated.

EXCEPTION_START

Every exception started.

FP_STATE

Floating point state. Fields:

FPSR unsigned int
: FPSR register value.

MASK unsigned int
: The mask indicating updated cumulative bits.

HLT

HLT instruction occurred. Fields:

IMM16 unsigned int
: The 16-bit immediate value encoded in HLT instruction.

INST

Every instruction executed. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DEBUG_STATE bool
: The instruction is executed in debug state.

DISASS string
: Disassembly of instruction.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The current instruction set.

ITSTATE unsigned int
: The current ITSTATE.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

MODE enum
: The mode the core is in.

NS unsigned int
: The core's non-secure bit.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

OPCODE unsigned int
: The opcode of the instruction.

PADDR unsigned int
: The physical address of the instruction.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the instruction.

PC unsigned int
: The address of the instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SIZE unsigned int
: The size of the instruction in bytes.

INST_END

Every instruction completed.

INST_START

Every instruction started. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

INST_STRADDLE

Instruction straddles boundary. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

ISB

ISB instruction occurred.

LOAD_MULTIPLE_REGS_LIST

List of destination registers (not source) of a multiple registers load instruction. Fields:

REG string
: Register to add into the list

LOCAL_MONITOR

Local monitor activity. Fields:

PADDR unsigned int
: Local Monitor Address

State enum
: State of the monitor (Open/Exclusive)

MEMMAP_DEBUG_READ

Memory mapped reads to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

MEMMAP_DEBUG_WRITE

Memory mapped writes to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

MMU_TRANS

Address translation information. Fields:

ASID unsigned int
: Address space identifier.

CORE_NUM unsigned int
: Core number in a multi processor.

Hyp bool
: Entry matches in Hyp state only

INNERCACHE_RA bool
: Is the inner cache allocate on read

INNERCACHE_TRANSIENT bool
: Is the inner write-through transient

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

MEMTYPE enum
: Memory type.

NSDESC enum
: Is secure side supposed to access secure or nonsecure memory

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TRANSIENT bool
: Is the outer write-through transient

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PADDR unsigned int
: Physical address of the access.

PAGESIZE unsigned int
: Page size as log2(size).

SH enum
: Shareability

SIDE enum
: Inst / Data.

STAGE1_PERM unsigned int
: Stage 1 permissions, [5:3] Privileged access (XWR) [2:0] User access (XWR)

STAGE2_PERM unsigned int
: Stage 2 permission mask [5:3] Privileged access (XWR) [2:0] User access (XWR)

VADDR unsigned int
: Virtual address of the access.

VMID unsigned int
: Virtual machine identifier.

XS bool
: XS Attribute

nG enum
: Flag indicating whether ASID will be matched

MMU_TTB_READ

This event is triggered by reads caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data read.

IPA unsigned int
: For stage 1, the IPA of the read.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

OUTADDR unsigned int
: The output physical address.

PADDR unsigned int
: The physical address of the read.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MMU_TTB_WRITE

This event is triggered by writes caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data written.

IPA unsigned int
: For stage 1, the IPA of the write.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

PADDR unsigned int
: The physical address of the write.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MODE_CHANGE

Mode change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: The new mode.

NON_SECURE enum
: The new security state

OLD_MODE enum
: The old mode.

NOT_TAG_CHECKED_INST

MTE Tag not checked for current instruction.

PERIODIC

Called for every quantum. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

PC unsigned int
: The address of the next instruction to be executed on this CPU.

PMU_COUNTER_OVERFLOW

PMU counter overflow. Fields:

EVENT_ID unsigned int
: Event Identifier

INDEX unsigned int
: Counter index (as selected by PMSELR).

INTERRUPT bool
: Is interrupt enabled on overflow for this counter

PRELOAD_DATA

Data preload from PLD instruction. Fields:

VADDR unsigned int
: Virtual address of the data that should be preloaded.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PRELOAD_INST

Instruction preload from PLI instruction. Fields:

VADDR unsigned int
: Virtual address of the instruction that should be preloaded.

PRE_CORE_LOAD

Trace just before a core load. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_CORE_STORE

Trace just before a core store. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_TTB_READ

This event is triggered before reads caused by a translation table walk. Fields:

ADDR unsigned int
: The physical address of the read.

PSTATE_PACTIVE_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

PSTATE_PACTIVE_OPMODE_TRANSITION_DENIED

Trace the denied pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_POWER_TRANSITION

Trace pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OLD_STATE enum
: Power mode before the transition

PSTATE_PACTIVE_POWER_TRANSITION_DENIED

Trace the denied pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_THREAD_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions for threads on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

RUN_STATE

Run state transition. Fields:

INST_COUNT unsigned int
: Ticks count at point of transition.

NEW enum
: New run state.

OLD enum
: Old run state.

SEMIHOSTING_CALL

Call of a semihost function occurred. Fields:

PC unsigned int
: The program counter after the semihosting call.

SEMIHOSTING_PRECALL

About to call semihost. Fields:

EL2 bool
: Use EL2 translation regime.

NS bool
: Is Non-Secure.

REG_WIDTH unsigned int
: The current register width in bytes.

SIGNAL

External signal state change. Fields:

SIGNAL enum
: Signal that changed

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

STATE bool
: Signal asserted state

SPSR

SPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: Which of the banked SPSR registers is written.

OLD_VALUE unsigned int
: The old SPSR value

VALUE unsigned int
: The new SPSR value

START_COMPILE

Compilation started. Fields:

VADDR unsigned int
: Instruction where compilation begins.

STG_MULTIPLE_MICRO_OPS_ADDRESS_RANGE

Address range for MTE stores with multiple uops. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

STORE_MULTIPLE_REGS_ADDRESS_RANGE

Address range for stores with multiple registers. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

SUBOPTIMAL_LDST_RETIRED

suboptimal load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1: half-word, 2: word ...)

OPTIMAL_ALIGN_SELECTOR unsigned int
: Selector to decide the optimal alignment size (1: access width, 2: 4-bytes, 3: 8-bytes, ..., 12: 4-kbytes)

VADDR unsigned int
: The virtual address of the access

SYNC

Called for every synchronization, that is every quantum break. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

LOCAL_QUANTUM unsigned int
: The local quantum of this CPU.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOCAL_TIME unsigned int
: The local time of this CPU.

SYNC_REASON enum
: Reason for sync.

SYSCALL

System call instruction executed. Fields:

IMM unsigned int
: Immediate value of the system call instruction.

TYPE enum
: System call type.

VADDR unsigned int
: Instruction that caused the system call.

SYSREG_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE32

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE64

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

UNALIGNED_LDST_RETIRED

Processor unaligned load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1=>half-word, 2->word ...)

NUMBER_OF_BEATS unsigned int
: Number of accesses (beats) in this burst (Total size in bytes is ACCESS_SIZE * NUMBER_OF_BEATS)

VADDR unsigned int
: The virtual address of the access

VFP_D_REGS

VFP/NEON D 64 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_Q_REGS

VFP/NEON Q 128 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding byte in the register. For example, 0x1 would indicate a write to VALUE[7:0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_SYS_REGS

Writes to the VFP/NEON units system registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: Which VFP system register is written.

MASK unsigned int
: The mask indicating updated cumulative bits.

OLD_VALUE unsigned int
: The register's old value overwritten.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VALUE unsigned int
: The new value written to the VFP system register.

VFP_S_REGS

VFP/NEON S 32 bit register write. Fields:

ALIASING enum
: Type of register aliasing used.

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

WAYPOINT

Signals end of basic block execution. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: Origin instructions set

IS_COND bool
: Indicates if this is a conditional waypoint.

PC unsigned int
: Origin address (or 0 if unavailable)

TAKEN bool
: Indicates if this waypoint was taken.

TARGET unsigned int
: Destination address

TARGET_ISET enum
: Destination instructions set

WFE_END

WFE ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFE.

WFE_EVENT_REGISTER

WFE event register status: set/clear, reason. Fields:

INST_COUNT unsigned int
: Ticks count.

REASON enum
: Reason for set/clear. Only REASON==0 (Cleared by WFE) clears the bit, all other reasons set it.

WFE_IGNORED

WFE ignored. Fields:

EVENT bool
: This WFE was ignored because the event register was set.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: Ticks count when ignoring WFE.

TRAPPED bool
: This WFE was trapped.

WFE_START

WFE entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFE.

WFI_END

WFI ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFI.

WFI_IGNORED

WFI ignored. Fields:

DISABLED bool
: This WFI was ignored because WFI is disabled.

INST_COUNT unsigned int
: Ticks count when ignoring WFI.

REASON enum
: specifies reason why WFI trace was ignored

TRAPPED bool
: This WFI was trapped.

WFI_START

WFI entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFI.

WFI_WAKEUP

WFI wakeup. Fields:

INST_COUNT unsigned int
: Ticks count when WFI wakeup occurred.

REASON enum
: Reason for wakeup.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.28 ARM_Cortex-A9UP - trace

This section describes the trace sources.

ASYNC_MEMORY_FAULT

Context ID Register write. Fields:

FAULT unsigned int
: Fault status in ESR format

PADDR unsigned int
: Physical Address (or 0 if unavailable)

VADDR signed int
: Virtual Address (or 0 if unavailable)

ATOMIC_END_ACCESS

Bus trace access for atomics. Fields:

ACCESS_FAIL bool
: Memory access failed

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

LOAD_VALUE unsigned int
: Loaded values

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ATOMIC_START_ACCESS

Bus trace access for atomics. Fields:

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

NSDESC unsigned int
: The security state of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ArchMsg.Error.exit_code

Fields:

EXITCODE signed int
: Exit code to be returned

KIND string
: Component kind that invoked the exit code trace

REASON string
: Reason why the exit code trace was invoked (optional)

ArchMsg.Warning.IgnoredWarmResetRequest

Event fires when software attempts to request warm reset through write to deprecated DBGPRCR.CWRR.

ArchMsg.Warning.TarmacReconstructor

Unexpected event happened during Tarmac reconstruction. Fields:

REASON string
: Reason

ArchMsg.Warning.Unpredictably Indexed PM Event Register

Fields:

IsDirect unsigned int
: Direct Access rather than indirect

N unsigned int
: Number of Accessible Registers

SEL unsigned int
: Selector

ArchMsg.Warning.branch_to_unaligned_address

Fields:

ADDR unsigned int
: Unaligned branch address

ArchMsg.Warning.cache_contents_unknown

A cache has been activated without being invalidated and may contain UNKNOWN content. Fields:

REGIME unsigned int
: Current translation regime

SIDE unsigned int
: 0 d-side cache or unified cache, 1 i-side

ArchMsg.Warning.dcimvac_matches_watchpoint

Watchpoints matching an AArch32 DCIMVA are implementation defined.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_fieldmismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_fracbitsnegative

VCVT Instruction is unpredictable with a negative number of fraction bits.

ArchMsg.Warning.decode_initblock

Instruction is unpredictable in an IT block.

ArchMsg.Warning.decode_initblocknotlast

Instruction is unpredictable in an IT block when not the last.

ArchMsg.Warning.decode_invalid_condition

Instruction is unpredictable when the condition is not AL.

ArchMsg.Warning.decode_invalidfieldcombination

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_invalidvalue

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_mem_hint_unallocated

There is no memory hint allocated to this bit pattern.

ArchMsg.Warning.decode_registermatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registermismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registeroutofrange

Float point instruction is unpredictable indexing beyond the end of the register bank.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_sbzsbo

Reserved bits in the instruction are not canonical.

ArchMsg.Warning.decode_transactiontoobig

Attempt to perform a memory transaction of more than 128 bytes.

ArchMsg.Warning.decode_unpred_other

Instruction is unpredictable.

ArchMsg.Warning.decode_unpreduseofpc

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13inreglist

Instruction is unpredictable with R13 in register-list.

ArchMsg.Warning.decode_useofd16orhigher

Attempt to use register D16 or higher where it is not allowed.

ArchMsg.Warning.decode_writebackandbaseinlist

Instruction is unpredictable with write back when the base register is in the transfer list.

ArchMsg.Warning.decode_zeroregistersinlist

Instruction is unpredictable when no registers are to be transfered.

ArchMsg.Warning.recursive_branch

An instruction is performing a branch that targets the same instruction. If this was intended then a WFI might be more effective. Fields:

PC unsigned int
: Address of instruction causing the branch

ArchMsg.Warning.recursive_exception

An instruction has generated an exception that targets the same instruction. Fields:

CPSR unsigned int
: Processor state

PC unsigned int
: Address of instruction causing the exception

TYPE enum
: Type of exception

ArchMsg.Warning.reentrant_vector_catch

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR unsigned int
vector address

ArchMsg.Warning.reserved_it_state

Fields:

ITSTATE unsigned int
reserved IT state

ArchMsg.Warning.tlb_contents_unknown

Translation has been enabled without invalidating the TLB. Fields:

EL unsigned int
Exception level of translation regime
INVALIDITY unsigned int
Which TLB may be invalid (if TLBs are separate)
NON_SECURE unsigned int
Security state of translation regime
VMID unsigned int
Current VMID, if applicable

ArchMsg.Warning.unpredictable_a32_breakpoint

Fields:

ADDR1 unsigned int
Address
ADDR2 unsigned int
Breakpoint address
BAS unsigned int
BAS Field
IS_ADDRESS_MISMATCH_BREAKPOINT bool
Is Mismatch breakpoint, not a match breakpoint
N unsigned int
Breakpoint number
OFFSET unsigned int
Indicates which halfword actually considering into the A32 instruction

ArchMsg.Warning.unpredictable_banked_register_access

Fields:

R unsigned int
R bit
SYSM unsigned int
SYSm

ArchMsg.Warning.unpredictable_ccfail_undef

Fields:

OPCODE unsigned int
opcode of the instruction
VADDR unsigned int
Virtual address of the instruction being executed

ArchMsg.Warning.unpredictable_ici_bits

ICI bits do not apply or are architecturally unpredictable for instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.unpredictable_t32_breakpoint

Fields:

ADDR1 unsigned int
: Address

ADDR2 unsigned int
: Breakpoint address

BAS unsigned int
: BAS Field

IS_ADDRESS_MISMATCH_BREAKPOINT bool
: Is Mismatch breakpoint, not a match breakpoint

N unsigned int
: Breakpoint number

OFFSET unsigned int
: Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_t32_vector_catch

Fields:

ADDR unsigned int
: Address

OFFSET unsigned int
: Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_watchpoint_far

Fields:

LOWER_BOUND unsigned int
: the lowest address accessed by the instruction that triggered the watchpoint

UPPER_BOUND unsigned int
: the highest watchpointed address accessed by that instruction

ArchMsg.Warning.warning_AdvSIMDExpandImmUnexpectedZero

AdvSIMDExpandImm may treat this immediate value as UNPREDICTABLE.

ArchMsg.Warning.warning_ConditionalSMC

SMC instruction has UNPREDICTABLE effects when conditional when combined with traps.

ArchMsg.Warning.warning_access_crosses_page_boundary

Access crosses page boundary.

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault_on_both_pages

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_spanning_different_memory

Fields:

ADDR unsigned int
: address of access crossing page

MEMTYPE_PAGE1 signed int
: First Page's memory type (0-Normal, 1-Non-Normal)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MEMTYPE_PAGE2 signed int
Second Page's memory type (0-Normal, 1-Non-Normal)

ArchMsg.Warning.warning_access_wraps_around_memory

Fields:

WIDTH unsigned int
Address width

ArchMsg.Warning.warning_bx_from_thumbee

Fields:

ADDR unsigned int
Destination address

ArchMsg.Warning.warning_ccsidr_unimplemented_level

Fields:

CSSELR unsigned int
current effective value of CSSELR

ArchMsg.Warning.warning_change_to_ns_when_tge_set

Attempting to change NS to 1 when HCR.TGE = 1.

ArchMsg.Warning.warning_contiguous_bit_check_abort

Fields:

ENTRY_ADDR unsigned int
address of conflicting TLB entry

ArchMsg.Warning.warning_contiguous_bit_error

Fields:

CONTIG_ADDR unsigned int
address of the TLB first read, with which this entry is expected to be contiguous
CONTIG_DATA unsigned int
expected contents based on the entry at CONTIG_ADDR
ENTRY_ADDR unsigned int
address of conflicting TLB entry
ENTRY_DATA unsigned int
contents of conflicting TLB entry

ArchMsg.Warning.warning_cp10_cp11_mismatch

Fields:

EL unsigned int
EL owning the control

ArchMsg.Warning.warning_cp10_cp11_reserved_value

Fields:

CP10 unsigned int
CP10 access
CP11 unsigned int
CP11 access

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_csselr_level_out_of_range

Fields:

IMPLEMENTED unsigned int
: number of cache levels implemented

LEVEL unsigned int
: written value of CSSELR.Level

TYPE enum
: The type of cache selected by CSSELR.InD and TnD

ArchMsg.Warning.warning_debug_flow_control_bits_not_obeyed

Fields:

ERROR enum
: type of error resulting

REGISTER enum
: register under consideration

ArchMsg.Warning.warning_debug_register_access_during_reset

Fields:

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

ArchMsg.Warning.warning_decode_cps_inconsistent_fields

The mode setting fields in the CPS instruction are inconsistent.

ArchMsg.Warning.warning_decode_invalid_state

Instruction is unpredictable in current exception-level/security state.

ArchMsg.Warning.warning_default_cacheable_mmu_on

Default cacheable is enabled (HCR.DC = 1) while MMU is enabled.

ArchMsg.Warning.warning_default_cacheable_vmmu_off

Default cacheable is enabled (HCR.DC = 1) while HCR.VM = 0.

ArchMsg.Warning.warning_deprecated_wvr_bit_2_set

Fields:

N unsigned int
: index

WVR unsigned int
: DBGWVR

ArchMsg.Warning.warning_eret_while_software_step_active_pending

Missing ISB between setting MDSCR_EL1.SS and ERET.

ArchMsg.Warning.warning_exclusive_to_non_normal

Fields:

ADDR unsigned int
: Address

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_exclusive_to_non_writeback

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_execute_from_device_memory

An attempt was made to execute from Device memory.

ArchMsg.Warning.warning_illegal_cpsr_mode

Fields:

NEW_MODE unsigned int
: New value of CPSR.M

ArchMsg.Warning.warning_illegal_srs_mode

Fields:

MODE unsigned int
: mode for Banked SP

ArchMsg.Warning.warning_implementation_defined_read_debug_register_in_SCS

Read Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_implementation_defined_sequential_security_transitions_supported

The behaviour of a sequential instruction fetches that cross from non-secure to secure memory and contain SG instruction is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_implementation_defined_stack_limit_check_supported

It is IMPLEMENTATION DEFINED whether stack pointer limit checking is performed for the this instructions. Fields:

ADDRESS unsigned int
: Address of instruction

OPCODE unsigned int
: instruction opcode

ArchMsg.Warning.warning_implementation_defined_write_debug_register_in_SCS

Write Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

DATA unsigned int
: data attempted to be written

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_invalid_tcr_granule

Fields:

REQUEST signed int
: page size requested (or 0 for reserved)

SUBSTITUTE signed int
: best guess available page size

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

TG_ID bool
bits TG0 or TG1

ArchMsg.Warning.warning_load_multiple_user_registers_from_user_mode

An LDM(user registers) instruction executed from user mode.

ArchMsg.Warning.warning_load_pc_from_unaligned

PC loaded from an unaligned location.

ArchMsg.Warning.warning_mte_in_mem_attr_not_supported

Fields:

ATTR unsigned int
Memory attribute

ArchMsg.Warning.warning_reserved_breakpoint_state_match

Fields:

IS_BREAKPOINT unsigned int
Is breakpoint, not watchpoint

ArchMsg.Warning.warning_shareability

Unpredictable: combination of the 1st and 2nd stages of translation is Normal Inner Non-Cacheable, Outer Non-Cacheable.

ArchMsg.Warning.warning_software_step_set_while_enabled

MDSCR_EL1.SS set to 1 while software step debug exceptions are enabled.

ArchMsg.Warning.warning_thumb_instruction_wraps_around_memory

Thumb instruction warps around memory.

ArchMsg.Warning.warning_ttbr_sbz_bits_are_not_zero

Fields:

MSB unsigned int
x-1
TTBR enum
which TTBR register

ArchMsg.Warning.warning_unaligned_address_dbgdtrrx_write

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_address_dbgdtrtx_read

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_device

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_strongly_ordered

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unknown_sau_rnr

SAU_RNR was set to an unsupported value.

ArchMsg.Warning.warning_unpredictable_AIRCR_PRIP_and_BFHFNMINP

The effect of setting both AIRCR.BFHFNMINP and AIRCR.PRIP to 1 is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTCLRACTIVE_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTCLRACTIVE if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_and_SYSRESETREQ

When the processor is halted in Debug state, if a write to the register writes a 1 to both VECTRESET and SYSRESETREQ, the behavior is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTRESET if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_incorrect_VKEY

The value 0x05FA must be written to AIRCR.VKEY, otherwise the register write is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_EXE_RETURN_Reserved_Bit

EXC_RETURN[23:7] are reserved with the special condition that all bits should be written as one. Values other than all 1s are UNPREDICTABLE. Fields:

EXC_RETURN unsigned int
Special PC value
SBO_MASK unsigned int
Which bits Should Be One to not UNPRED

ArchMsg.Warning.warning_unpredictable_change_to_DEMCR_MON_STEP_at_insufficient_priority

The effect of changing DEMCR.MON_STEP at an execution priority that is lower than the priority of the DebugMonitor exception is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_change_to_priority_of_active_exception

Changing the priority of an active exception is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_clear_lspact

Fields:

DRVEXC_TAKEN unsigned int
derived exception taken not pended

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

REACHED_LAST_STORE unsigned int
only the last faulted

ArchMsg.Warning.warning_unpredictable_dbg_exit_unaligned_dlr

Debug exit to AArch32 with DLR[0] set to 1. PC[0] can be set to 0 or to DLR[0].

ArchMsg.Warning.warning_unpredictable_exception_catch

Generation of an exception catch event is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_inconsistent_state

state on exception return is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_instruction

exception return instruction is unpredictable.

ArchMsg.Warning.warning_unpredictable_in_debug_state

Instruction is UNPREDICTABLE when executed in debug state.

ArchMsg.Warning.warning_unpredictable_mair_encoding

Fields:

ATTR unsigned int
Memory attribute

ArchMsg.Warning.warning_unpredictable_pmu_counter_access

Access to PMU counters from non-secure EL0 or EL1 is UNPREDICTABLE with MDCR_EL2.HPMN set to 0.

ArchMsg.Warning.warning_unpredictable_prioritization_breakpoint_match_vector_catch

Two events with the same priority occurred at the same instruction: Address Matching Vector Catch debug event, and Breakpoint debug event It is constrained unpredictable which is taken.

ArchMsg.Warning.warning_unpredictable_stack_selection

UNPREDICTABLE state on context switch.

ArchMsg.Warning.warning_unpredictable_tsize_out_of_range

Fields:

TCR enum
Which TCR
TSIZE signed int
Value of TSize
TSIZE_MAX signed int
The maximum allowed value
TTBR unsigned int
Which TTBR region

ArchMsg.Warning.warning_unpredictable_unaligned_pc_as_base_register

The PC value must be word-aligned, otherwise the behavior of the instruction is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_unaligned_pop_stack

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

FRAME_ADDR unsigned int
: unaligned stack frame address

ArchMsg.Warning.warning_unpredictable_vmsa_memattrib

Fields:

TEXCB unsigned int
: texcb value

TEX_REMAP bool
: use tex remap

ArchMsg.Warning.warning_unpredictable_vtcr_t0sz_sl0

Fields:

SL0 signed int
: VTCR.SL0, starting level for stage 2 translation table walks

T0SZ signed int
: VTCR.T0SZ, required input address range

ArchMsg.Warning.warning_unpredictable_write_DHCSR

Fields:

NEW unsigned int
: data attempted to be written

OLD unsigned int
: previous value

ArchMsg.Warning.warning_unpredictable_write_SHCSR

UNPREDICTABLE setting a pending bit to 1 for an exception with priority >= execution priority.

ArchMsg.Warning.warning_unsupported_access_to_memory_mapped_register

Access to memory mapped register is unsupported.

ArchMsg.Warning.warning_user_jmcr_access

Fields:

WRITE unsigned int
: access is write

ArchMsg.Warning.warning_wcr_mask_and_bas

DBGWCR_EL1.MASK is non-zero and BAS is not 0xff.

ArchMsg.Warning.warning_wcr_mask_reserved

DBGWCR_ELn.MASK is set to a reserved value.

ArchMsg.Warning.warning_wcr_non_configuous_bas

DBGWCR_EL1.BAS has non-contiguous set of ones.

ArchMsg.Warning.warning_wvr_masked_bit_not_zero

DBGWVRn_EL1.VA contains a bit masked by DBGWCRn_EL1.MASK that is not zero.

BRANCH_MISPREDICT

Simulating branch mispredict. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PC unsigned int
Origin address (or 0 if unavailable)

BRA_DIR

Direct branches, to immediate address. Fields:

CORE_NUM unsigned int
Core number in a multi processor.
INST_COUNT unsigned int
The core's instruction counter, starting at 1 for the first instruction.
ISET enum
The instructions set of the branch instruction.
IS_COND bool
Indicates if this is a conditional branch.
IS_LINK bool
Indicates if this is a branch with link.
PC unsigned int
The address of the branch instruction.
TARGET_ISET enum
The instructions set after the branch.
TARGET_PC unsigned int
The address the instruction branches to.

BRA_INDIR

Indirect branches, perhaps to a register. Fields:

CORE_NUM unsigned int
Core number in a multi processor.
INST_COUNT unsigned int
The core's instruction counter, starting at 1 for the first instruction.
ISET enum
The instructions set of the branch instruction.
IS_COND bool
Indicates if this is a conditional branch.
IS_LINK bool
Indicates if this is a branch with link.
PC unsigned int
The address of the branch instruction.
TARGET_ISET enum
The instructions set after the branch.
TARGET_PC unsigned int
The address the instruction branches to.

BRA_INDIR_CCFAIL

Indirect branches fail, perhaps to a register.

BRA_RET_CCFAIL

Branche return fail, perhaps to a register.

CACHE_MAINTENANCE_OP

Cache Maintenance Operation. Fields:

CORE_NUM unsigned int
Core number in a multi processor.
DATA unsigned int
Specified MVA or Set/way

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

FUNCTION enum
: Clean / Invalidate

SCOPE enum
: Affected region

SIDE enum
: Inst / Data.

CCFAIL

Conditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCFAIL_UNC

Unconditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS

Conditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS_UNC

Unconditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CODE_CACHE_MAINT

Code cache maintenance. Fields:

END_INCL unsigned int
: Inclusive end addres for by VA or PA requests

NS bool
: Non-secure world for PA

START unsigned int
: Start address for by VA or PA requests

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

TYPE enum
: Request type.

COMPILE_BLOCK_END

Last instruction of basic block translated. Fields:

VADDR unsigned int
: Address of next instruction after this basic block

COMPILE_BLOCK_START

First instruction of basic block translated. Fields:

VADDR unsigned int
: Address of first instruction in basic block

COMPILE_INST

ARM instruction compiled. Fields:

DISASS string
: Disassembly of the instruction.

ISET enum
: The instruction set of this instruction.

ITSTATE unsigned int
: The ITSTATE current for the instruction.

OPCODE unsigned int
: The opcode of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

CONTEXTIDR

Context ID Register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

NS bool
: Secure or nonsecure banked register is accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

CONTEXT_SYNC

Called for every context synchronization event. Fields:

ARCH_CSE bool
: Architectural Context Synchronization Event

SYSREG_SYNC bool
: System register synchronization event

CORE_ENDIAN

Core BE8 Big-Endian state changed. Fields:

BE8 bool
: Core BE8 Big-Endian state

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_INFO

Static processor attributes. Only triggered by a call to DumpState(). Fields:

ARCH_PROFILE enum
: The architecture profile of the core.

CLUSTER_ID unsigned int
: The cluster ID of this processor.

CORE_NUM unsigned int
: The number of this core in an MP processor.

DETAILED_CORE_FEATURES enum
: Detailed core features as a bitfield. Enum values are bitmasks to extract the info from the field

FPU_VERSION enum
: The VFP version implemented by the core.

MEM_ARCH enum
: The memory architecture of the core.

MIDR_EL1 unsigned int
: MIDR_EL1 value

NUM_CORES unsigned int
: The number of cores in this MP processor.

QUANTUM_SIZE unsigned int
: The default quantum size of the core.

SECURITY_FEATURES bool
: Does the core have security features?

CORE_LOADS

Processor load accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CORE_REGS

Changes of the core registers R0 to R14. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number, 0 to 14.

MODE enum
: Bank of the register accessed.

OLD_VALUE unsigned int
: The old value overwritten.

PHYS_ID enum
: The physical register accessed.

VALUE unsigned int
: The new value written to the register.

CORE_STORES

Processor store accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CP14_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP14_WRITE

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

CP15_WRITE

System Control Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

CPSR

CPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

OLD_VALUE unsigned int
: The old CPSR value

UNKNOWN unsigned int
: Bits within the register that have unknown value.

VALUE unsigned int
: The new CPSR value

CRYPTO_SPEC

Every crypto instruction speculatively executed.

CorePowerStateChange

Core power-up state has changed. Fields:

PU bool
: Whether the core powered up

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DATA_CACHE_SET_ALLOC_TAGS

Set allocation tags from DC GVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of memory in bytes.

TAG unsigned int
: The value of the tag being set.

VADDR unsigned int
: Virtual address used for setting allocation tags.

DATA_CACHE_ZERO

Zero and invalidate cache line from DC ZVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of zero'd memory in bytes.

VADDR unsigned int
: Virtual address of the zero'd cache line.

DEBUG_EVENT

Hardware debug support event. Fields:

EVENT enum
: Description of event

VALUE unsigned int
: data value

END_COMPILE

Compilation end. Fields:

END_OF_PAGE_COUNT unsigned int
: Number of basic blocks exited due to page boundary since START_COMPILE.

FETCHFAIL_COUNT unsigned int
: Number of basic blocks exited due to ifetch failure START_COMPILE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: Number of instructions compiled since START_COMPILE.

NONSEQ_COUNT unsigned int
: Number of basic blocks exited due to non-sequential instructions since START_COMPILE.

PAGE_STRADDLE_COUNT unsigned int
: Number of basic blocks exited due to unaligned instructions crossing page since START_COMPILE.

EXCEPTION

Exception taken. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

IS_PHYSICAL bool
: Physical or Virtual exception

LR unsigned int
: The value assigned to the link register.

PC unsigned int
: The location where the exception occurred.

PREFERRED_RETURN unsigned int
: The preferred return address for the exception.

TARGET_ISET enum
: The instruction set of the exception handler code.

TARGET_PC unsigned int
: The address the exception branches to.

VECTOR enum
: The exception vector.

VECTOR_OFFSET unsigned int
: The offset of an exception vector from the base address.

EXCEPTION_END

Every exception completed.

EXCEPTION_RAISE

Every exception raised.

EXCEPTION_RETURN

Branches on leaving exception. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

EXCEPTION_RETURN_PREBRANCH

Exception return event that must occur before PC and processor state have updated.

EXCEPTION_START

Every exception started.

FP_STATE

Floating point state. Fields:

FPSR `unsigned int`
: FPSR register value.

MASK `unsigned int`
: The mask indicating updated cumulative bits.

HLT

HLT instruction occurred. Fields:

IMM16 `unsigned int`
: The 16-bit immediate value encoded in HLT instruction.

INST

Every instruction executed. Fields:

CORE_NUM `unsigned int`
: Core number in a multi processor.

CURRENT_TIME `unsigned int`
: The core's current time, as simulated time plus local time.

DEBUG_STATE `bool`
: The instruction is executed in debug state.

DISASS `string`
: Disassembly of instruction.

INST_COUNT `unsigned int`
: The core's instruction counter, starting at 1 for the first instruction.

ISET `enum`
: The current instruction set.

ITSTATE `unsigned int`
: The current ITSTATE.

LOCAL_TIME `unsigned int`
: The core's local time, relative to the current quantum.

MODE `enum`
: The mode the core is in.

NS `unsigned int`
: The core's non-secure bit.

NSDESC `unsigned int`
: The physical address non-secure bit.

NSDESC2 `unsigned int`
: The second page physical address non-secure bit.

OPCODE `unsigned int`
: The opcode of the instruction.

PADDR `unsigned int`
: The physical address of the instruction.

PADDR2 `unsigned int`
: If different from PADDR, the physical address of the second page of the instruction.

PC `unsigned int`
: The address of the instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SIZE unsigned int
: The size of the instruction in bytes.

INST_END

Every instruction completed.

INST_START

Every instruction started. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

INST_STRADDLE

Instruction straddles boundary. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

ISB

ISB instruction occurred.

LOAD_MULTIPLE_REGS_LIST

List of destination registers (not source) of a multiple registers load instruction. Fields:

REG string
: Register to add into the list

LOCAL_MONITOR

Local monitor activity. Fields:

PADDR unsigned int
: Local Monitor Address

State enum
: State of the monitor (Open/Exclusive)

MEMMAP_DEBUG_READ

Memory mapped reads to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

MEMMAP_DEBUG_WRITE

Memory mapped writes to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

MMU_TRANS

Address translation information. Fields:

ASID unsigned int
: Address space identifier.

CORE_NUM unsigned int
: Core number in a multi processor.

Hyp bool
: Entry matches in Hyp state only

INNERCACHE_RA bool
: Is the inner cache allocate on read

INNERCACHE_TRANSIENT bool
: Is the inner write-through transient

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

MEMTYPE enum
: Memory type.

NSDESC enum
: Is secure side supposed to access secure or nonsecure memory

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TRANSIENT bool
: Is the outer write-through transient

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PADDR unsigned int
: Physical address of the access.

PAGESIZE unsigned int
: Page size as log2(size).

SH enum
: Shareability

SIDE enum
: Inst / Data.

STAGE1_PERM unsigned int
: Stage 1 permissions, [5:3] Privileged access (XWR) [2:0] User access (XWR)

STAGE2_PERM unsigned int
: Stage 2 permission mask [5:3] Privileged access (XWR) [2:0] User access (XWR)

VADDR unsigned int
: Virtual address of the access.

VMID unsigned int
: Virtual machine identifier.

XS bool
: XS Attribute

nG enum
: Flag indicating whether ASID will be matched

MMU_TTB_READ

This event is triggered by reads caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data read.

IPA unsigned int
: For stage 1, the IPA of the read.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

OUTADDR unsigned int
: The output physical address.

PADDR unsigned int
: The physical address of the read.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MMU_TTB_WRITE

This event is triggered by writes caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data written.

IPA unsigned int
: For stage 1, the IPA of the write.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

PADDR unsigned int
: The physical address of the write.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MODE_CHANGE

Mode change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: The new mode.

NON_SECURE enum
: The new security state

OLD_MODE enum
: The old mode.

NOT_TAG_CHECKED_INST

MTE Tag not checked for current instruction.

PERIODIC

Called for every quantum. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

PC unsigned int
: The address of the next instruction to be executed on this CPU.

PMU_COUNTER_OVERFLOW

PMU counter overflow. Fields:

EVENT_ID unsigned int
: Event Identifier

INDEX unsigned int
: Counter index (as selected by PMSELR).

INTERRUPT bool
: Is interrupt enabled on overflow for this counter

PRELOAD_DATA

Data preload from PLD instruction. Fields:

VADDR unsigned int
: Virtual address of the data that should be preloaded.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PRELOAD_INST

Instruction preload from PLI instruction. Fields:

VADDR unsigned int
: Virtual address of the instruction that should be preloaded.

PRE_CORE_LOAD

Trace just before a core load. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_CORE_STORE

Trace just before a core store. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_TTB_READ

This event is triggered before reads caused by a translation table walk. Fields:

ADDR unsigned int
: The physical address of the read.

PSTATE_PACTIVE_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

PSTATE_PACTIVE_OPMODE_TRANSITION_DENIED

Trace the denied pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_POWER_TRANSITION

Trace pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OLD_STATE enum
: Power mode before the transition

PSTATE_PACTIVE_POWER_TRANSITION_DENIED

Trace the denied pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_THREAD_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions for threads on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

RUN_STATE

Run state transition. Fields:

INST_COUNT unsigned int
: Ticks count at point of transition.

NEW enum
: New run state.

OLD enum
: Old run state.

SEMIHOSTING_CALL

Call of a semihost function occurred. Fields:

PC unsigned int
: The program counter after the semihosting call.

SEMIHOSTING_PRECALL

About to call semihost. Fields:

EL2 bool
: Use EL2 translation regime.

NS bool
: Is Non-Secure.

REG_WIDTH unsigned int
: The current register width in bytes.

SIGNAL

External signal state change. Fields:

SIGNAL enum
: Signal that changed

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

STATE bool
: Signal asserted state

SPSR

SPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: Which of the banked SPSR registers is written.

OLD_VALUE unsigned int
: The old SPSR value

VALUE unsigned int
: The new SPSR value

START_COMPILE

Compilation started. Fields:

VADDR unsigned int
: Instruction where compilation begins.

STG_MULTIPLE_MICRO_OPS_ADDRESS_RANGE

Address range for MTE stores with multiple uops. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

STORE_MULTIPLE_REGS_ADDRESS_RANGE

Address range for stores with multiple registers. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

SUBOPTIMAL_LDST_RETIRED

suboptimal load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1: half-word, 2: word ...)

OPTIMAL_ALIGN_SELECTOR unsigned int
: Selector to decide the optimal alignment size (1: access width, 2: 4-bytes, 3: 8-bytes, ..., 12: 4-kbytes)

VADDR unsigned int
: The virtual address of the access

SYNC

Called for every synchronization, that is every quantum break. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

LOCAL_QUANTUM unsigned int
: The local quantum of this CPU.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOCAL_TIME unsigned int
: The local time of this CPU.

SYNC_REASON enum
: Reason for sync.

SYSCALL

System call instruction executed. Fields:

IMM unsigned int
: Immediate value of the system call instruction.

TYPE enum
: System call type.

VADDR unsigned int
: Instruction that caused the system call.

SYSREG_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE32

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE64

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

UNALIGNED_LDST_RETIRED

Processor unaligned load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1=>half-word, 2->word ...)

NUMBER_OF_BEATS unsigned int
: Number of accesses (beats) in this burst (Total size in bytes is ACCESS_SIZE * NUMBER_OF_BEATS)

VADDR unsigned int
: The virtual address of the access

VFP_D_REGS

VFP/NEON D 64 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_Q_REGS

VFP/NEON Q 128 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding byte in the register. For example, 0x1 would indicate a write to VALUE[7:0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_SYS_REGS

Writes to the VFP/NEON units system registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: Which VFP system register is written.

MASK unsigned int
: The mask indicating updated cumulative bits.

OLD_VALUE unsigned int
: The register's old value overwritten.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VALUE unsigned int
: The new value written to the VFP system register.

VFP_S_REGS

VFP/NEON S 32 bit register write. Fields:

ALIASING enum
: Type of register aliasing used.

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

WAYPOINT

Signals end of basic block execution. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: Origin instructions set

IS_COND bool
: Indicates if this is a conditional waypoint.

PC unsigned int
: Origin address (or 0 if unavailable)

TAKEN bool
: Indicates if this waypoint was taken.

TARGET unsigned int
: Destination address

TARGET_ISET enum
: Destination instructions set

WFE_END

WFE ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFE.

WFE_EVENT_REGISTER

WFE event register status: set/clear, reason. Fields:

INST_COUNT unsigned int
: Ticks count.

REASON enum
: Reason for set/clear. Only REASON==0 (Cleared by WFE) clears the bit, all other reasons set it.

WFE_IGNORED

WFE ignored. Fields:

EVENT bool
: This WFE was ignored because the event register was set.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: Ticks count when ignoring WFE.

TRAPPED bool
: This WFE was trapped.

WFE_START

WFE entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFE.

WFI_END

WFI ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFI.

WFI_IGNORED

WFI ignored. Fields:

DISABLED bool
: This WFI was ignored because WFI is disabled.

INST_COUNT unsigned int
: Ticks count when ignoring WFI.

REASON enum
: specifies reason why WFI trace was ignored

TRAPPED bool
: This WFI was trapped.

WFI_START

WFI entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFI.

WFI_WAKEUP

WFI wakeup. Fields:

INST_COUNT unsigned int
: Ticks count when WFI wakeup occurred.

REASON enum
: Reason for wakeup.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.29 ARM_Cortex-M0 - trace

This section describes the trace sources.

ASYNC_MEMORY_FAULT

Context ID Register write. Fields:

FAULT unsigned int
: Fault status

PADDR unsigned int
: Physical Address (or 0 if unavailable)

ATOMIC_END_ACCESS

Bus trace access for atomics. Fields:

ACCESS_FAIL bool
: Memory access failed

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

LOAD_VALUE unsigned int
: Loaded values

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ATOMIC_START_ACCESS

Bus trace access for atomics. Fields:

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ArchMsg.Error.exit_code

Fields:

EXITCODE signed int
: Exit code to be returned

KIND string
: Component kind that invoked the exit code trace

REASON string
: Reason why the exit code trace was invoked (optional)

ArchMsg.Warning.TarmacReconstructor

Unexpected event happened during Tarmac reconstruction. Fields:

REASON string
: Reason

ArchMsg.Warning.Unpredictably Indexed PM Event Register

Fields:

IsDirect unsigned int
: Direct Access rather than indirect

N unsigned int
: Number of Accessible Registers

SEL unsigned int
: Selector

ArchMsg.Warning.branch_to_unaligned_address

Fields:

ADDR unsigned int
: Unaligned branch address

ArchMsg.Warning.dap_csw_bad_size

A write to CM3DAP CSW has an invalid size field. Fields:

DATA unsigned int
: bits[2:0] are size

ArchMsg.Warning.dcimvac_matches_watchpoint

Watchpoints matching an AArch32 DCIMVA are implementation defined.

ArchMsg.Warning.decode_fieldmismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_fracbitsnegative

VCVT Instruction is unpredictable with a negative number of fraction bits.

ArchMsg.Warning.decode_initblock

Instruction is unpredictable in an IT block.

ArchMsg.Warning.decode_initblocknotlast

Instruction is unpredictable in an IT block when not the last.

ArchMsg.Warning.decode_invalid_condition

Instruction is unpredictable when the condition is not AL.

ArchMsg.Warning.decode_invalidfieldcombination

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_invalidvalue

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_mem_hint_unallocated

There is no memory hint allocated to this bit pattern.

ArchMsg.Warning.decode_registermatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registermismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registeroutofrange

Float point instruction is unpredictable indexing beyond the end of the register bank.

ArchMsg.Warning.decode_sbzsbo

Reserved bits in the instruction are not canonical.

ArchMsg.Warning.decode_transactiontoobig

Attempt to perform a memory transaction of more than 128 bytes.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_unpred_other

Instruction is unpredictable.

ArchMsg.Warning.decode_unpreduseofpc

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13inreglist

Instruction is unpredictable with R13 in register-list.

ArchMsg.Warning.decode_useofd16orhigher

Attempt to use register D16 or higher where it is not allowed.

ArchMsg.Warning.decode_writebackandbaseinlist

Instruction is unpredictable with write back when the base register is in the transfer list.

ArchMsg.Warning.decode_zeroregistersinlist

Instruction is unpredictable when no registers are to be transfered.

ArchMsg.Warning.recursive_branch

An instruction is performing a branch that targets the same instruction. If this was intended then a WFI might be more effective. Fields:

PC unsigned int
: Address of instruction causing the branch

ArchMsg.Warning.recursive_exception

An instruction has generated an exception that targets the same instruction. Fields:

CPSR unsigned int
: Processor state

PC unsigned int
: Address of instruction causing the exception

TYPE enum
: Type of exception

ArchMsg.Warning.reentrant_vector_catch

Fields:

ADDR unsigned int
: vector address

ArchMsg.Warning.reserved_it_state

Fields:

ITSTATE unsigned int
: reserved IT state

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.secure_vector_fetch_from_nonsecure

Secure vector table is being fetched from Non-secure memory .

ArchMsg.Warning.sg_in_it_block

Arm recommends that software does not place SG (security gate instruction) inside an IT instruction block.

ArchMsg.Warning.unpredictable_a32_breakpoint

Fields:

ADDR1 unsigned int
: Address

ADDR2 unsigned int
: Breakpoint address

BAS unsigned int
: BAS Field

IS_ADDRESS_MISMATCH_BREAKPOINT bool
: Is Mismatch breakpoint, not a match breakpoint

N unsigned int
: Breakpoint number

OFFSET unsigned int
: Indicates which halfword actually considering into the A32 instruction

ArchMsg.Warning.unpredictable_banked_register_access

Fields:

R unsigned int
: R bit

SYSM unsigned int
: SYSm

ArchMsg.Warning.unpredictable_ccfail_undef

Fields:

OPCODE unsigned int
: opcode of the instruction

VADDR unsigned int
: Virtual address of the instruction being executed

ArchMsg.Warning.unpredictable_ici_bits

ICI bits do not apply or are architecturally unpredictable for instruction.

ArchMsg.Warning.unpredictable_t32_breakpoint

Fields:

ADDR1 unsigned int
: Address

ADDR2 unsigned int
: Breakpoint address

BAS unsigned int
: BAS Field

IS_ADDRESS_MISMATCH_BREAKPOINT bool
: Is Mismatch breakpoint, not a match breakpoint

N unsigned int
: Breakpoint number

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_t32_vector_catch

Fields:

ADDR unsigned int
Address
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_watchpoint_far

Fields:

LOWER_BOUND unsigned int
the lowest address accessed by the instruction that triggered the watchpoint
UPPER_BOUND unsigned int
the highest watchpointed address accessed by that instruction

ArchMsg.Warning.warning_AdvSIMDExpandImmUnexpectedZero

AdvSIMDExpandImm may treat this immediate value as UNPREDICTABLE.

ArchMsg.Warning.warning_ConditionalSMC

SMC instruction has UNPREDICTABLE effects when conditional when combined with traps.

ArchMsg.Warning.warning_access_crosses_page_boundary

Access crosses page boundary.

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault_on_both_pages

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_spanning_different_memory

Fields:

ADDR unsigned int
address of access crossing page
MEMTYPE_PAGE1 signed int
First Page's memory type (0-Normal, 1-Non-Normal)
MEMTYPE_PAGE2 signed int
Second Page's memory type (0-Normal, 1-Non-Normal)

ArchMsg.Warning.warning_access_wraps_around_memory

Fields:

WIDTH unsigned int
Address width

ArchMsg.Warning.warning_bx_from_thumbee

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR unsigned int
Destination address

ArchMsg.Warning.warning_ccsidr_unimplemented_level

Fields:

CSSELR unsigned int
current effective value of CSSELR

ArchMsg.Warning.warning_change_to_ns_when_tge_set

Attempting to change NS to 1 when HCR.TGE = 1.

ArchMsg.Warning.warning_contiguous_bit_check_abort

Fields:

ENTRY_ADDR unsigned int
address of conflicting TLB entry

ArchMsg.Warning.warning_contiguous_bit_error

Fields:

CONTIG_ADDR unsigned int
address of the TLB first read, with which this entry is expected to be contiguous
CONTIG_DATA unsigned int
expected contents based on the entry at CONTIG_ADDR
ENTRY_ADDR unsigned int
address of conflicting TLB entry
ENTRY_DATA unsigned int
contents of conflicting TLB entry

ArchMsg.Warning.warning_cp10_cp11_mismatch

Fields:

EL unsigned int
EL owning the control

ArchMsg.Warning.warning_cp10_cp11_reserved_value

Fields:

CP10 unsigned int
CP10 access
CP11 unsigned int
CP11 access

ArchMsg.Warning.warning_csselr_level_out_of_range

Fields:

IMPLEMENTED unsigned int
number of cache levels implemented
LEVEL unsigned int
written value of CSSELR.Level
TYPE enum
The type of cache selected by CSSELR.InD and TnD

ArchMsg.Warning.warning_debug_flow_control_bits_not_obeyed

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ERROR enum
: type of error resulting

REGISTER enum
: register under consideration

ArchMsg.Warning.warning_debug_register_access_during_reset

Fields:

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

ArchMsg.Warning.warning_decode_cps_inconsistent_fields

The mode setting fields in the CPS instruction are inconsistent.

ArchMsg.Warning.warning_decode_invalid_state

Instruction is unpredictable in current exception-level/security state.

ArchMsg.Warning.warning_default_cacheable_mmu_on

Default cacheable is enabled (HCR.DC = 1) while MMU is enabled.

ArchMsg.Warning.warning_default_cacheable_vmmu_off

Default cacheable is enabled (HCR.DC = 1) while HCR.VM = 0.

ArchMsg.Warning.warning_deprecated_wvr_bit_2_set

Fields:

N unsigned int
: index

WVR unsigned int
: DBGWVR

ArchMsg.Warning.warning_eret_while_software_step_active_pending

Missing ISB between setting MDSCR_EL1.SS and ERET.

ArchMsg.Warning.warning_exclusive_to_non_normal

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_exclusive_to_non_writeback

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_execute_from_device_memory

An attempt was made to execute from Device memory.

ArchMsg.Warning.warning_illegal_cpsr_mode

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NEW_MODE unsigned int
New value of CPSR.M

ArchMsg.Warning.warning_illegal_srs_mode

Fields:

MODE unsigned int
mode for Banked SP

ArchMsg.Warning.warning_implementation_defined_read_debug_register_in_SCS

Read Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

PPB_OFFSET unsigned int
debug register offset

ArchMsg.Warning.warning_implementation_defined_sequential_security_transitions_supported

The behaviour of a sequential instruction fetches that cross from non-secure to secure memory and contain SG instruction is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_implementation_defined_stack_limit_check_supported

It is IMPLEMENTATION DEFINED whether stack pointer limit checking is performed for the this instructions. Fields:

ADDRESS unsigned int
Address of instruction
OPCODE unsigned int
instruction opcode

ArchMsg.Warning.warning_implementation_defined_write_debug_register_in_SCS

Write Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

DATA unsigned int
data attempted to be written
PPB_OFFSET unsigned int
debug register offset

ArchMsg.Warning.warning_invalid_tcr_granule

Fields:

REQUEST signed int
page size requested (or 0 for reserved)
SUBSTITUTE signed int
best guess available page size
TG_ID bool
bits TG0 or TG1

ArchMsg.Warning.warning_load_multiple_user_registers_from_user_mode

An LDM(user registers) instruction executed from user mode.

ArchMsg.Warning.warning_load_pc_from_unaligned

PC loaded from an unaligned location.

ArchMsg.Warning.warning_mte_in_mem_attr_not_supported

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ATTR unsigned int
Memory attribute

ArchMsg.Warning.warning_reserved_breakpoint_state_match

Fields:

IS_BREAKPOINT unsigned int
Is breakpoint, not watchpoint

ArchMsg.Warning.warning_shareability

Unpredictable: combination of the 1st and 2nd stages of translation is Normal Inner Non-Cacheable, Outer Non-Cacheable.

ArchMsg.Warning.warning_software_step_set_while_enabled

MDSCR_EL1.SS set to 1 while software step debug exceptions are enabled.

ArchMsg.Warning.warning_thumb_instruction_wraps_around_memory

Thumb instruction warps around memory.

ArchMsg.Warning.warning_ttbr_sbz_bits_are_not_zero

Fields:

MSB unsigned int
x-1
TTBR enum
which TTBR register

ArchMsg.Warning.warning_unaligned_address_dbgdtrrx_write

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_address_dbgdtrtx_read

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_device

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_strongly_ordered

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unknown_sau_rnr

SAU_RNR was set to an unsupported value.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_AIRCR_PRIP_and_BFHFNMINP

The effect of setting both AIRCR.BFHFNMINP and AIRCR.PRIP to 1 is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTCLRACTIVE_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTCLRACTIVE if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_and_SYSRESETREQ

When the processor is halted in Debug state, if a write to the register writes a 1 to both VECTRESET and SYSRESETREQ, the behavior is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTRESET if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_incorrect_VKEY

The value 0x05FA must be written to AIRCR.VKEY, otherwise the register write is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_EXE_RETURN_Reserved_Bit

EXC_RETURN[23:7] are reserved with the special condition that all bits should be written as one. Values other than all 1s are UNPREDICTABLE. Fields:

EXC_RETURN `unsigned int`
: Special PC value

SBO_MASK `unsigned int`
: Which bits Should Be One to not UNPRED

ArchMsg.Warning.warning_unpredictable_change_to_DEMCR_MON_STEP_at_insufficient_priority

The effect of changing DEMCR.MON_STEP at an execution priority that is lower than the priority of the DebugMonitor exception is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_change_to_priority_of_active_exception

Changing the priority of an active exception is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_clear_lspact

Fields:

DRVEXC_TAKEN `unsigned int`
: derived exception taken not pended

REACHED_LAST_STORE `unsigned int`
: only the last faulted

ArchMsg.Warning.warning_unpredictable_dbg_exit_unaligned_dlr

Debug exit to AArch32 with DLR[0] set to 1. PC[0] can be set to 0 or to DLR[0].

ArchMsg.Warning.warning_unpredictable_exception_catch

Generation of an exception catch event is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_inconsistent_state

state on exception return is unpredictable.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_exception_return_instruction

exception return instruction is unpredictable.

ArchMsg.Warning.warning_unpredictable_in_debug_state

Instruction is UNPREDICTABLE when executed in debug state.

ArchMsg.Warning.warning_unpredictable_mair_encoding

Fields:

ATTR unsigned int
: Memory attribute

ArchMsg.Warning.warning_unpredictable_pmu_counter_access

Access to PMU counters from non-secure EL0 or EL1 is UNPREDICTABLE with MDCR_EL2.HPMN set to 0.

ArchMsg.Warning.warning_unpredictable_prioritization_breakpoint_match_vector_catch

Two events with the same priority occurred at the same instruction: Address Matching Vector Catch debug event, and Breakpoint debug event It is constrained unpredictable which is taken.

ArchMsg.Warning.warning_unpredictable_stack_selection

UNPREDICTABLE state on context switch.

ArchMsg.Warning.warning_unpredictable_tsize_out_of_range

Fields:

TCR enum
: Which TCR

TSIZE signed int
: Value of TSize

TSIZE_MAX signed int
: The maximum allowed value

TTBR unsigned int
: Which TTBR region

ArchMsg.Warning.warning_unpredictable_unaligned_pc_as_base_register

The PC value must be word-aligned, otherwise the behavior of the instruction is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_unaligned_pop_stack

Fields:

FRAME_ADDR unsigned int
: unaligned stack frame address

ArchMsg.Warning.warning_unpredictable_vmsa_memattrib

Fields:

TEXCB unsigned int
: texcb value

TEX_REMAP bool
: use tex remap

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_vtcr_t0sz_sl0

Fields:

SL0 signed int
: VTCR.SL0, starting level for stage 2 translation table walks

T0SZ signed int
: VTCR.T0SZ, required input address range

ArchMsg.Warning.warning_unpredictable_write_DHCSR

Fields:

NEW unsigned int
: data attempted to be written

OLD unsigned int
: previous value

ArchMsg.Warning.warning_unpredictable_write_SHCSR

UNPREDICTABLE setting a pending bit to 1 for an exception with priority >= execution priority.

ArchMsg.Warning.warning_unsupported_access_to_memory_mapped_register

Access to memory mapped register is unsupported.

ArchMsg.Warning.warning_user_jmcr_access

Fields:

WRITE unsigned int
: access is write

ArchMsg.Warning.warning_wcr_mask_and_bas

DBGWCR_EL1.MASK is non-zero and BAS is not 0xff.

ArchMsg.Warning.warning_wcr_mask_reserved

DBGWCR_ELn.MASK is set to a reserved value.

ArchMsg.Warning.warning_wcr_non_configuous_bas

DBGWCR_EL1.BAS has non-contiguous set of ones.

ArchMsg.Warning.warning_wvr_masked_bit_not_zero

DBGWVRn_EL1.VA contains a bit masked by DBGWCRn_EL1.MASK that is not zero.

ArchMsg.warning_unpred_system_register_access

Fields:

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

BRANCH_MISPREDICT

Simulating branch mispredict. Fields:

PC unsigned int
: Origin address (or 0 if unavailable)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

BRA_DIR

Direct branches, to immediate address. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR

Indirect branches, perhaps to a register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR_CCFAIL

Indirect branches fail, perhaps to a register.

BRA_RET_CCFAIL

Branche return fail, perhaps to a register.

CCFAIL

Conditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CCFAIL_UNC

Unconditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS

Conditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS_UNC

Unconditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CODE_CACHE_MAINT

Code cache maintenance. Fields:

END_INCL unsigned int
: Inclusive end addres for by VA or PA requests

NS bool
: Non-secure world for PA

START unsigned int
: Start address for by VA or PA requests

TYPE enum
: Request type.

COMPILE_BLOCK_END

Last instruction of basic block translated. Fields:

VADDR unsigned int
: Address of next instruction after this basic block

COMPILE_BLOCK_START

First instruction of basic block translated. Fields:

VADDR unsigned int
: Address of first instruction in basic block

COMPILE_INST

ARM instruction compiled. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DISASS string
: Disassembly of the instruction.

ISET enum
: The instruction set of this instruction.

ITSTATE unsigned int
: The ITSTATE current for the instruction.

OPCODE unsigned int
: The opcode of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

CONTEXTIDR

Context ID Register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

NS bool
: Secure or nonsecure banked register is accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

CONTEXT_SYNC

Called for every context synchronization event. Fields:

ARCH_CSE bool
: Architectural Context Synchronization Event

SYSREG_SYNC bool
: System register synchronization event

CORE_ENDIAN

Core BE8 Big-Endian state changed. Fields:

BE8 bool
: Core BE8 Big-Endian state

CORE_INFO

Static processor attributes. Only triggered by a call to DumpState(). Fields:

ARCH_PROFILE enum
: The architecture profile of the core.

CLUSTER_ID unsigned int
: The cluster ID of this processor.

CORE_NUM unsigned int
: The number of this core in an MP processor.

DETAILED_CORE_FEATURES enum
: Detailed core features as a bitfield. Enum values are bitmasks to extract the info from the field

FPU_VERSION enum
: The VFP version implemented by the core.

MEM_ARCH enum
: The memory architecture of the core.

MIDR_EL1 unsigned int
: MIDR_EL1 value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NUM_CORES unsigned int
: The number of cores in this MP processor.

QUANTUM_SIZE unsigned int
: The default quantum size of the core.

SECURITY_FEATURES bool
: Does the core have security features?

CORE_LOADS

Processor load accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CORE_REGS

Changes of the core registers R0 to R14. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ID unsigned int
: The register number, 0 to 14.

OLD_VALUE unsigned int
: The old value overwritten.

PHYS_ID enum
: The physical register accessed.

VALUE unsigned int
: The new value written to the register.

CORE_STORES

Processor store accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CRYPTO_SPEC

Every crypto instruction speculatively executed.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DATA_CACHE_SET_ALLOC_TAGS

Set allocation tags from DC GVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of memory in bytes.

TAG unsigned int
: The value of the tag being set.

VADDR unsigned int
: Virtual address used for setting allocation tags.

DATA_CACHE_ZERO

Zero and invalidate cache line from DC ZVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of zero'd memory in bytes.

VADDR unsigned int
: Virtual address of the zero'd cache line.

DEBUG_EVENT

Hardware debug support event. Fields:

EVENT enum
: Description of event

VALUE unsigned int
: data value

DWT_MATCH

DWT comparator matches. Fields:

NUM unsigned int
: DWT comparator number

TYPE enum
: DWT comparator configuration

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

END_COMPILE

Compilation end. Fields:

END_OF_PAGE_COUNT unsigned int
: Number of basic blocks exited due to page boundary since START_COMPILE.

FETCHFAIL_COUNT unsigned int
: Number of basic blocks exited due to ifetch failure START_COMPILE.

INST_COUNT unsigned int
: Number of instructions compiled since START_COMPILE.

NONSEQ_COUNT unsigned int
: Number of basic blocks exited due to non-sequential instructions since START_COMPILE.

PAGE_STRADDLE_COUNT unsigned int
: Number of basic blocks exited due to unaligned instructions crossing page since START_COMPILE.

EXCEPTION

Exceptions which are taken. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: The instruction set of the processor when the exception occurred.

LR unsigned int
: The value assigned to the link register.

PC unsigned int
: The location where the exception occurred.

TARGET_ISET enum
: The instruction set of the exception handler code.

TARGET_PC unsigned int
: The address the exception branches to.

VECTOR enum
: The exception vector.

EXCEPTION_END

Every exception completed.

EXCEPTION_ENTRY

Event marking the entry of an exception. All pushes of register to the stack will follow this event. Fields:

PC unsigned int
: The location where the exception occurred.

VECTOR enum
: The exception that occurred.

EXCEPTION_RAISE

Every exception raised.

EXCEPTION_RETURN

Marks the end of an exception, but on an M core does not indicate a program flow change (branch). In most situations it will be followed by an indirect branch. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ISET enum
: The instructions set of the branch instruction.

PC unsigned int
: The address of the branch instruction.

EXCEPTION_RETURN_PREBRANCH

Exception return event that must occur before PC and processor state have updated.

EXCEPTION_START

Every exception started.

EXCEPTION_VECTOR_FETCH

I-side vector fetch for M-class exception. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Result of fetch if OK

RESPONSE enum
: 0=Aborted, 1=OK

VADDR unsigned int
: Where the vector is fetched from (including VTOR)

FP_STATE

Floating point state. Fields:

FPSR unsigned int
: FPSR register value.

MASK unsigned int
: The mask indicating updated cumulative bits.

HLT

HLT instruction occurred. Fields:

IMM16 unsigned int
: The 16-bit immediate value encoded in HLT instruction.

INFO_EXCEPTION_REASON

Provide information of real cause of exception. Fields:

FaultCause enum
: Which bit of which V7M/V8M-mainline FSR/HFSR would have been set.

PC unsigned int
: The location where the exception occurred.

PHASE enum
: What we are doing with the exception.

REASONS enum
: Why we are doing it.

VECTOR enum
: The exception that occurred.

INFO_STACKING

Provide information of real cause of exception. Fields:

PHASE enum
: Are we starting or ending a sequence of stack writes.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

REASONS enum
: Why we are doing it.

INST

Every instruction executed. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DEBUG_STATE bool
: The instruction is executed in debug state.

DISASS string
: Disassembly of instruction.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The current instruction set.

ITSTATE unsigned int
: The current ITSTATE.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

MODE enum
: The mode the core is in.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

OPCODE unsigned int
: The opcode of the instruction.

PADDR unsigned int
: The physical address of the instruction.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

INST_END

Every instruction completed.

INST_START

Every instruction started. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

INST_STRADDLE

Instruction straddles boundary. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

ISB

ISB instruction occurred.

LOAD_MULTIPLE_REGS_LIST

List of destination registers (not source) of a multiple registers load instruction. Fields:

REG string
: Register to add into the list

LOCAL_MONITOR

Local monitor activity. Fields:

PADDR unsigned int
: Local Monitor Address

State enum
: State of the monitor (Open/Exclusive)

LOCKUP_CYCLE

This event is triggered if the core enters or stays in the lock-up state. Fields:

CAUSE enum
: What causes the lockup?.

LOCKUP_ENTRY

This event is triggered if the core enters the lock-up state. Fields:

PC unsigned int
: The PC when the lockup state is entered.

LOCKUP_EXIT

This event is triggered if the core leaves the lock-up state.

MODE_CHANGE

Mode change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: The new mode.

OLD_MODE enum
: The old mode.

MPU_TRANS

Address translation information. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

EXEC_PERM enum
: Execution Permission

INNERCACHE_RA bool
: Is the inner cache allocate on read

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

MEMTYPE enum
: Memory type.

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

PADDR unsigned int
: Address of the access.

PAGESIZE unsigned int
: Page size as log2(size).

READ_PERM enum
: Read Permission

REG_NUM signed int
: MPU region-number 0-15 (else 0xFF=none, 0xBB/0xDD=background/default)

SH enum
: Shareability

SIDE enum
: Inst / Data.

WRITE_PERM enum
: Write Permission

PERIODIC

Called for every quantum. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

PC unsigned int
: The address of the next instruction to be executed on this CPU.

PPB_READ_ACCESS

Trace reads from the core to memory mapped registers in the PPB (private peripheral bus) address range. Fields:

ACCESS_SIZE unsigned int
: Logarithm of the access size: 0=byte, 1=halfword, 2=word, 3=doubleword, ...

BURST_LEN unsigned int
: The number of data transfers (beats) in this burst.

DATA unsigned int
: The data transferred.

INNER_ATTRIBUTES unsigned int
: The inner attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

INST bool
: Is an instruction fetch.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOCK enum
: Normal / Locked / Exclusive.

MASTER_ID unsigned int
: The AXI master ID.

NS bool
: Is a nonsecure access.

OUTER_ATTRIBUTES unsigned int
: The outer attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

PADDR unsigned int
: Physical address of access.

PRIV bool
: Is privileged access.

RESPONSE enum
: Whether the transaction was successful, or an error occurred.

USER_FLAGS unsigned int
: Core specific additional signals.

PPB_WRITE_ACCESS

Trace writes from the core to memory mapped registers in the PPB (private peripheral bus) address range. Fields:

ACCESS_SIZE unsigned int
: Logarithm of the access size: 0=byte, 1=halfword, 2=word, 3=doubleword, ...

BURST_LEN unsigned int
: The number of data transfers (beats) in this burst.

DATA unsigned int
: The data transferred.

INNER_ATTRIBUTES unsigned int
: The inner attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

INST bool
: Is an instruction fetch.

LOCK enum
: Normal / Locked / Exclusive.

MASTER_ID unsigned int
: The AXI master ID.

NS bool
: Is a nonsecure access.

OUTER_ATTRIBUTES unsigned int
: The outer attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

PADDR unsigned int
: Physical address of access.

PRIV bool
: Is privileged access.

RESPONSE enum
: Whether the transaction was successful, or an error occurred.

USER_FLAGS unsigned int
: Core specific additional signals.

PRELOAD_DATA

Data preload from PLD instruction. Fields:

VADDR unsigned int
: Virtual address of the data that should be preloaded.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PRELOAD_INST

Instruction preload from PLI instruction. Fields:

VADDR unsigned int
: Virtual address of the instruction that should be preloaded.

PRE_CORE_LOAD

Trace just before a core load. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_CORE_STORE

Trace just before a core store. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PSTATE_PACTIVE_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

PSTATE_PACTIVE_OPMODE_TRANSITION_DENIED

Trace the denied pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_POWER_TRANSITION

Trace pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

PSTATE_PACTIVE_POWER_TRANSITION_DENIED

Trace the denied pactive and power mode transitions on PChannel. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_THREAD_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions for threads on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

RUN_STATE

Run state transition. Fields:

INST_COUNT unsigned int
: Ticks count at point of transition.

NEW enum
: New run state.

OLD enum
: Old run state.

SEMIHOSTING_CALL

Call of a semihost function occurred. Fields:

PC unsigned int
: The program counter after the semihosting call.

SEMIHOSTING_PRECALL

About to call semihost. Fields:

EL2 bool
: Use EL2 translation regime.

NS bool
: Is Non-Secure.

REG_WIDTH unsigned int
: The current register width in bytes.

SIGNAL

External signal state change. Fields:

SIGNAL enum
: Signal that changed

STATE bool
: Signal asserted state

START_COMPILE

Compilation started. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VADDR unsigned int
: Instruction where compilation begins.

STG_MULTIPLE_MICRO_OPS_ADDRESS_RANGE

Address range for MTE stores with multiple uops. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

STORE_MULTIPLE_REGS_ADDRESS_RANGE

Address range for stores with multiple registers. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

SUBOPTIMAL_LDST_RETIRED

suboptimal load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1: half-word, 2: word ...)

OPTIMAL_ALIGN_SELECTOR unsigned int
: Selector to decide the optimal alignment size (1: access width, 2: 4-bytes, 3: 8-bytes, ..., 12: 4-kbytes)

VADDR unsigned int
: The virtual address of the access

SYNC

Called for every synchronization, that is every quantum break. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

LOCAL_QUANTUM unsigned int
: The local quantum of this CPU.

LOCAL_TIME unsigned int
: The local time of this CPU.

SYNC_REASON enum
: Reason for sync.

SYSCALL

System call instruction executed. Fields:

IMM unsigned int
: Immediate value of the system call instruction.

TYPE enum
: System call type.

VADDR unsigned int
: Instruction that caused the system call.

SYSREG_READ

Trace reads from system registers. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

REG_NAME enum
: Register number

VALUE unsigned int
: The value read.

SYSREG_UPDATE32

Triggers when the system updates a register. Fields:

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

SYSREG_UPDATE64

Triggers when the system updates a register. Fields:

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

SYSREG_WRITE

Trace write to system registers. Fields:

REG_ACCESS enum
: Register access status

REG_NAME enum
: Register number

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: Value written to the register.

UNALIGNED_LDST_RETIRED

Processor unaligned load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1=>half-word, 2->word ...)

NUMBER_OF_BEATS unsigned int
: Number of accesses (beats) in this burst (Total size in bytes is ACCESS_SIZE * NUMBER_OF_BEATS)

VADDR unsigned int
: The virtual address of the access

WAYPOINT

Signals end of basic block execution. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: Origin instructions set

IS_COND bool
: Indicates if this is a conditional waypoint.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PC unsigned int
: Origin address (or 0 if unavailable)

TAKEN bool
: Indicates if this waypoint was taken.

TARGET unsigned int
: Destination address

TARGET_ISET enum
: Destination instructions set

WFE_END

WFE ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFE.

WFE_EVENT_REGISTER

WFE event register status: set/clear, reason. Fields:

INST_COUNT unsigned int
: Ticks count.

REASON enum
: Reason for set/clear. Only REASON==0 (Cleared by WFE) clears the bit, all other reasons set it.

WFE_IGNORED

WFE ignored. Fields:

EVENT bool
: This WFE was ignored because the event register was set.

INST_COUNT unsigned int
: Ticks count when ignoring WFE.

TRAPPED bool
: This WFE was trapped.

WFE_START

WFE entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFE.

WFI_END

WFI ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFI.

WFI_IGNORED

WFI ignored. Fields:

DISABLED bool
: This WFI was ignored because WFI is disabled.

INST_COUNT unsigned int
: Ticks count when ignoring WFI.

REASON enum
: specifies reason why WFI trace was ignored

TRAPPED bool
: This WFI was trapped.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

WFI_START

WFI entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFI.

WFI_WAKEUP

WFI wakeup. Fields:

INST_COUNT unsigned int
: Ticks count when WFI wakeup occurred.

REASON enum
: Reason for wakeup.

XPSR

Changes to the xPSR register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

OLD_VALUE unsigned int
: The old xPSR value

UNKNOWN unsigned int
: Bits within the register that have unknown value.

VALUE unsigned int
: The new xPSR value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.30 ARM_Cortex-M0+ - trace

This section describes the trace sources.

ASYNC_MEMORY_FAULT

Context ID Register write. Fields:

FAULT unsigned int
: Fault status

PADDR unsigned int
: Physical Address (or 0 if unavailable)

ATOMIC_END_ACCESS

Bus trace access for atomics. Fields:

ACCESS_FAIL bool
: Memory access failed

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

LOAD_VALUE unsigned int
: Loaded values

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ATOMIC_START_ACCESS

Bus trace access for atomics. Fields:

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ArchMsg.Error.exit_code

Fields:

EXITCODE signed int
: Exit code to be returned

KIND string
: Component kind that invoked the exit code trace

REASON string
: Reason why the exit code trace was invoked (optional)

ArchMsg.Warning.TarmacReconstructor

Unexpected event happened during Tarmac reconstruction. Fields:

REASON string
: Reason

ArchMsg.Warning.Unpredictably Indexed PM Event Register

Fields:

IsDirect unsigned int
: Direct Access rather than indirect

N unsigned int
: Number of Accessible Registers

SEL unsigned int
: Selector

ArchMsg.Warning.branch_to_unaligned_address

Fields:

ADDR unsigned int
: Unaligned branch address

ArchMsg.Warning.dap_csw_bad_size

A write to CM3DAP CSW has an invalid size field. Fields:

DATA unsigned int
: bits[2:0] are size

ArchMsg.Warning.dcimvac_matches_watchpoint

Watchpoints matching an AArch32 DCIMVA are implementation defined.

ArchMsg.Warning.decode_fieldmismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_fracbitsnegative

VCVT Instruction is unpredictable with a negative number of fraction bits.

ArchMsg.Warning.decode_initblock

Instruction is unpredictable in an IT block.

ArchMsg.Warning.decode_initblocknotlast

Instruction is unpredictable in an IT block when not the last.

ArchMsg.Warning.decode_invalid_condition

Instruction is unpredictable when the condition is not AL.

ArchMsg.Warning.decode_invalidfieldcombination

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_invalidvalue

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_mem_hint_unallocated

There is no memory hint allocated to this bit pattern.

ArchMsg.Warning.decode_registermatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registermismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registeroutofrange

Float point instruction is unpredictable indexing beyond the end of the register bank.

ArchMsg.Warning.decode_sbzsbo

Reserved bits in the instruction are not canonical.

ArchMsg.Warning.decode_transactiontoobig

Attempt to perform a memory transaction of more than 128 bytes.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_unpred_other

Instruction is unpredictable.

ArchMsg.Warning.decode_unpreduseofpc

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13inreglist

Instruction is unpredictable with R13 in register-list.

ArchMsg.Warning.decode_useofd16orhigher

Attempt to use register D16 or higher where it is not allowed.

ArchMsg.Warning.decode_writebackandbaseinlist

Instruction is unpredictable with write back when the base register is in the transfer list.

ArchMsg.Warning.decode_zeroregistersinlist

Instruction is unpredictable when no registers are to be transfered.

ArchMsg.Warning.recursive_branch

An instruction is performing a branch that targets the same instruction. If this was intended then a WFI might be more effective. Fields:

PC unsigned int
: Address of instruction causing the branch

ArchMsg.Warning.recursive_exception

An instruction has generated an exception that targets the same instruction. Fields:

CPSR unsigned int
: Processor state

PC unsigned int
: Address of instruction causing the exception

TYPE enum
: Type of exception

ArchMsg.Warning.reentrant_vector_catch

Fields:

ADDR unsigned int
: vector address

ArchMsg.Warning.reserved_it_state

Fields:

ITSTATE unsigned int
: reserved IT state

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.secure_vector_fetch_from_nonsecure

Secure vector table is being fetched from Non-secure memory .

ArchMsg.Warning.sg_in_it_block

Arm recommends that software does not place SG (security gate instruction) inside an IT instruction block.

ArchMsg.Warning.unpredictable_a32_breakpoint

Fields:

ADDR1 unsigned int
: Address

ADDR2 unsigned int
: Breakpoint address

BAS unsigned int
: BAS Field

IS_ADDRESS_MISMATCH_BREAKPOINT bool
: Is Mismatch breakpoint, not a match breakpoint

N unsigned int
: Breakpoint number

OFFSET unsigned int
: Indicates which halfword actually considering into the A32 instruction

ArchMsg.Warning.unpredictable_banked_register_access

Fields:

R unsigned int
: R bit

SYSM unsigned int
: SYSm

ArchMsg.Warning.unpredictable_ccfail_undef

Fields:

OPCODE unsigned int
: opcode of the instruction

VADDR unsigned int
: Virtual address of the instruction being executed

ArchMsg.Warning.unpredictable_ici_bits

ICI bits do not apply or are architecturally unpredictable for instruction.

ArchMsg.Warning.unpredictable_t32_breakpoint

Fields:

ADDR1 unsigned int
: Address

ADDR2 unsigned int
: Breakpoint address

BAS unsigned int
: BAS Field

IS_ADDRESS_MISMATCH_BREAKPOINT bool
: Is Mismatch breakpoint, not a match breakpoint

N unsigned int
: Breakpoint number

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_t32_vector_catch

Fields:

ADDR unsigned int
Address
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_watchpoint_far

Fields:

LOWER_BOUND unsigned int
the lowest address accessed by the instruction that triggered the watchpoint
UPPER_BOUND unsigned int
the highest watchpointed address accessed by that instruction

ArchMsg.Warning.warning_AdvSIMDExpandImmUnexpectedZero

AdvSIMDExpandImm may treat this immediate value as UNPREDICTABLE.

ArchMsg.Warning.warning_ConditionalSMC

SMC instruction has UNPREDICTABLE effects when conditional when combined with traps.

ArchMsg.Warning.warning_access_crosses_page_boundary

Access crosses page boundary.

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault_on_both_pages

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_spanning_different_memory

Fields:

ADDR unsigned int
address of access crossing page
MEMTYPE_PAGE1 signed int
First Page's memory type (0-Normal, 1-Non-Normal)
MEMTYPE_PAGE2 signed int
Second Page's memory type (0-Normal, 1-Non-Normal)

ArchMsg.Warning.warning_access_wraps_around_memory

Fields:

WIDTH unsigned int
Address width

ArchMsg.Warning.warning_bx_from_thumbee

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR unsigned int
Destination address

ArchMsg.Warning.warning_ccsidr_unimplemented_level

Fields:

CSSELR unsigned int
current effective value of CSSELR

ArchMsg.Warning.warning_change_to_ns_when_tge_set

Attempting to change NS to 1 when HCR.TGE = 1.

ArchMsg.Warning.warning_contiguous_bit_check_abort

Fields:

ENTRY_ADDR unsigned int
address of conflicting TLB entry

ArchMsg.Warning.warning_contiguous_bit_error

Fields:

CONTIG_ADDR unsigned int
address of the TLB first read, with which this entry is expected to be contiguous
CONTIG_DATA unsigned int
expected contents based on the entry at CONTIG_ADDR
ENTRY_ADDR unsigned int
address of conflicting TLB entry
ENTRY_DATA unsigned int
contents of conflicting TLB entry

ArchMsg.Warning.warning_cp10_cp11_mismatch

Fields:

EL unsigned int
EL owning the control

ArchMsg.Warning.warning_cp10_cp11_reserved_value

Fields:

CP10 unsigned int
CP10 access
CP11 unsigned int
CP11 access

ArchMsg.Warning.warning_csselr_level_out_of_range

Fields:

IMPLEMENTED unsigned int
number of cache levels implemented
LEVEL unsigned int
written value of CSSELR.Level
TYPE enum
The type of cache selected by CSSELR.InD and TnD

ArchMsg.Warning.warning_debug_flow_control_bits_not_obeyed

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ERROR enum
: type of error resulting

REGISTER enum
: register under consideration

ArchMsg.Warning.warning_debug_register_access_during_reset

Fields:

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

ArchMsg.Warning.warning_decode_cps_inconsistent_fields

The mode setting fields in the CPS instruction are inconsistent.

ArchMsg.Warning.warning_decode_invalid_state

Instruction is unpredictable in current exception-level/security state.

ArchMsg.Warning.warning_default_cacheable_mmu_on

Default cacheable is enabled (HCR.DC = 1) while MMU is enabled.

ArchMsg.Warning.warning_default_cacheable_vmmu_off

Default cacheable is enabled (HCR.DC = 1) while HCR.VM = 0.

ArchMsg.Warning.warning_deprecated_wvr_bit_2_set

Fields:

N unsigned int
: index

WVR unsigned int
: DBGWVR

ArchMsg.Warning.warning_eret_while_software_step_active_pending

Missing ISB between setting MDSCR_EL1.SS and ERET.

ArchMsg.Warning.warning_exclusive_to_non_normal

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_exclusive_to_non_writeback

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_execute_from_device_memory

An attempt was made to execute from Device memory.

ArchMsg.Warning.warning_illegal_cpsr_mode

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NEW_MODE unsigned int
New value of CPSR.M

ArchMsg.Warning.warning_illegal_srs_mode

Fields:

MODE unsigned int
mode for Banked SP

ArchMsg.Warning.warning_implementation_defined_read_debug_register_in_SCS

Read Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

PPB_OFFSET unsigned int
debug register offset

ArchMsg.Warning.warning_implementation_defined_sequential_security_transitions_supported

The behaviour of a sequential instruction fetches that cross from non-secure to secure memory and contain SG instruction is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_implementation_defined_stack_limit_check_supported

It is IMPLEMENTATION DEFINED whether stack pointer limit checking is performed for the this instructions. Fields:

ADDRESS unsigned int
Address of instruction
OPCODE unsigned int
instruction opcode

ArchMsg.Warning.warning_implementation_defined_write_debug_register_in_SCS

Write Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

DATA unsigned int
data attempted to be written
PPB_OFFSET unsigned int
debug register offset

ArchMsg.Warning.warning_invalid_tcr_granule

Fields:

REQUEST signed int
page size requested (or 0 for reserved)
SUBSTITUTE signed int
best guess available page size
TG_ID bool
bits TG0 or TG1

ArchMsg.Warning.warning_load_multiple_user_registers_from_user_mode

An LDM(user registers) instruction executed from user mode.

ArchMsg.Warning.warning_load_pc_from_unaligned

PC loaded from an unaligned location.

ArchMsg.Warning.warning_mte_in_mem_attr_not_supported

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ATTR unsigned int
Memory attribute

ArchMsg.Warning.warning_reserved_breakpoint_state_match

Fields:

IS_BREAKPOINT unsigned int
Is breakpoint, not watchpoint

ArchMsg.Warning.warning_shareability

Unpredictable: combination of the 1st and 2nd stages of translation is Normal Inner Non-Cacheable, Outer Non-Cacheable.

ArchMsg.Warning.warning_software_step_set_while_enabled

MDSCR_EL1.SS set to 1 while software step debug exceptions are enabled.

ArchMsg.Warning.warning_thumb_instruction_wraps_around_memory

Thumb instruction warps around memory.

ArchMsg.Warning.warning_ttbr_sbz_bits_are_not_zero

Fields:

MSB unsigned int
x-1
TTBR enum
which TTBR register

ArchMsg.Warning.warning_unaligned_address_dbgdtrrx_write

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_address_dbgdtrtx_read

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_device

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_strongly_ordered

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unknown_sau_rnr

SAU_RNR was set to an unsupported value.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_AIRCR_PRIP_and_BFHFNMINP

The effect of setting both AIRCR.BFHFNMINP and AIRCR.PRIP to 1 is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTCLRACTIVE_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTCLRACTIVE if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_and_SYSRESETREQ

When the processor is halted in Debug state, if a write to the register writes a 1 to both VECTRESET and SYSRESETREQ, the behavior is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTRESET if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_incorrect_VKEY

The value 0x05FA must be written to AIRCR.VKEY, otherwise the register write is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_EXE_RETURN_Reserved_Bit

EXC_RETURN[23:7] are reserved with the special condition that all bits should be written as one. Values other than all 1s are UNPREDICTABLE. Fields:

EXC_RETURN `unsigned int`
: Special PC value

SBO_MASK `unsigned int`
: Which bits Should Be One to not UNPRED

ArchMsg.Warning.warning_unpredictable_change_to_DEMCR_MON_STEP_at_insufficient_priority

The effect of changing DEMCR.MON_STEP at an execution priority that is lower than the priority of the DebugMonitor exception is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_change_to_priority_of_active_exception

Changing the priority of an active exception is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_clear_lspact

Fields:

DRVEXC_TAKEN `unsigned int`
: derived exception taken not pended

REACHED_LAST_STORE `unsigned int`
: only the last faulted

ArchMsg.Warning.warning_unpredictable_dbg_exit_unaligned_dlr

Debug exit to AArch32 with DLR[0] set to 1. PC[0] can be set to 0 or to DLR[0].

ArchMsg.Warning.warning_unpredictable_exception_catch

Generation of an exception catch event is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_inconsistent_state

state on exception return is unpredictable.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_exception_return_instruction

exception return instruction is unpredictable.

ArchMsg.Warning.warning_unpredictable_in_debug_state

Instruction is UNPREDICTABLE when executed in debug state.

ArchMsg.Warning.warning_unpredictable_mair_encoding

Fields:

ATTR unsigned int
: Memory attribute

ArchMsg.Warning.warning_unpredictable_pmu_counter_access

Access to PMU counters from non-secure EL0 or EL1 is UNPREDICTABLE with MDCR_EL2.HPMN set to 0.

ArchMsg.Warning.warning_unpredictable_prioritization_breakpoint_match_vector_catch

Two events with the same priority occurred at the same instruction: Address Matching Vector Catch debug event, and Breakpoint debug event It is constrained unpredictable which is taken.

ArchMsg.Warning.warning_unpredictable_stack_selection

UNPREDICTABLE state on context switch.

ArchMsg.Warning.warning_unpredictable_tsize_out_of_range

Fields:

TCR enum
: Which TCR

TSIZE signed int
: Value of TSize

TSIZE_MAX signed int
: The maximum allowed value

TTBR unsigned int
: Which TTBR region

ArchMsg.Warning.warning_unpredictable_unaligned_pc_as_base_register

The PC value must be word-aligned, otherwise the behavior of the instruction is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_unaligned_pop_stack

Fields:

FRAME_ADDR unsigned int
: unaligned stack frame address

ArchMsg.Warning.warning_unpredictable_vmsa_memattrib

Fields:

TEXCB unsigned int
: texcb value

TEX_REMAP bool
: use tex remap

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_vtcr_t0sz_sl0

Fields:

SL0 signed int
: VTCR.SL0, starting level for stage 2 translation table walks

T0SZ signed int
: VTCR.T0SZ, required input address range

ArchMsg.Warning.warning_unpredictable_write_DHCSR

Fields:

NEW unsigned int
: data attempted to be written

OLD unsigned int
: previous value

ArchMsg.Warning.warning_unpredictable_write_SHCSR

UNPREDICTABLE setting a pending bit to 1 for an exception with priority >= execution priority.

ArchMsg.Warning.warning_unsupported_access_to_memory_mapped_register

Access to memory mapped register is unsupported.

ArchMsg.Warning.warning_user_jmcr_access

Fields:

WRITE unsigned int
: access is write

ArchMsg.Warning.warning_wcr_mask_and_bas

DBGWCR_EL1.MASK is non-zero and BAS is not 0xff.

ArchMsg.Warning.warning_wcr_mask_reserved

DBGWCR_ELn.MASK is set to a reserved value.

ArchMsg.Warning.warning_wcr_non_configuous_bas

DBGWCR_EL1.BAS has non-contiguous set of ones.

ArchMsg.Warning.warning_wvr_masked_bit_not_zero

DBGWVRn_EL1.VA contains a bit masked by DBGWCRn_EL1.MASK that is not zero.

ArchMsg.warning_unpred_system_register_access

Fields:

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

BRANCH_MISPREDICT

Simulating branch mispredict. Fields:

PC unsigned int
: Origin address (or 0 if unavailable)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

BRA_DIR

Direct branches, to immediate address. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR

Indirect branches, perhaps to a register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR_CCFAIL

Indirect branches fail, perhaps to a register.

BRA_RET_CCFAIL

Branche return fail, perhaps to a register.

CCFAIL

Conditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CCFAIL_UNC

Unconditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS

Conditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS_UNC

Unconditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CODE_CACHE_MAINT

Code cache maintenance. Fields:

END_INCL unsigned int
: Inclusive end addres for by VA or PA requests

NS bool
: Non-secure world for PA

START unsigned int
: Start address for by VA or PA requests

TYPE enum
: Request type.

COMPILE_BLOCK_END

Last instruction of basic block translated. Fields:

VADDR unsigned int
: Address of next instruction after this basic block

COMPILE_BLOCK_START

First instruction of basic block translated. Fields:

VADDR unsigned int
: Address of first instruction in basic block

COMPILE_INST

ARM instruction compiled. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DISASS string
: Disassembly of the instruction.

ISET enum
: The instruction set of this instruction.

ITSTATE unsigned int
: The ITSTATE current for the instruction.

OPCODE unsigned int
: The opcode of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

CONTEXTIDR

Context ID Register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

NS bool
: Secure or nonsecure banked register is accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

CONTEXT_SYNC

Called for every context synchronization event. Fields:

ARCH_CSE bool
: Architectural Context Synchronization Event

SYSREG_SYNC bool
: System register synchronization event

CORE_ENDIAN

Core BE8 Big-Endian state changed. Fields:

BE8 bool
: Core BE8 Big-Endian state

CORE_INFO

Static processor attributes. Only triggered by a call to DumpState(). Fields:

ARCH_PROFILE enum
: The architecture profile of the core.

CLUSTER_ID unsigned int
: The cluster ID of this processor.

CORE_NUM unsigned int
: The number of this core in an MP processor.

DETAILED_CORE_FEATURES enum
: Detailed core features as a bitfield. Enum values are bitmasks to extract the info from the field

FPU_VERSION enum
: The VFP version implemented by the core.

MEM_ARCH enum
: The memory architecture of the core.

MIDR_EL1 unsigned int
: MIDR_EL1 value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NUM_CORES unsigned int
: The number of cores in this MP processor.

QUANTUM_SIZE unsigned int
: The default quantum size of the core.

SECURITY_FEATURES bool
: Does the core have security features?

CORE_LOADS

Processor load accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CORE_REGS

Changes of the core registers R0 to R14. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ID unsigned int
: The register number, 0 to 14.

OLD_VALUE unsigned int
: The old value overwritten.

PHYS_ID enum
: The physical register accessed.

VALUE unsigned int
: The new value written to the register.

CORE_STORES

Processor store accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CRYPTO_SPEC

Every crypto instruction speculatively executed.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DATA_CACHE_SET_ALLOC_TAGS

Set allocation tags from DC GVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of memory in bytes.

TAG unsigned int
: The value of the tag being set.

VADDR unsigned int
: Virtual address used for setting allocation tags.

DATA_CACHE_ZERO

Zero and invalidate cache line from DC ZVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of zero'd memory in bytes.

VADDR unsigned int
: Virtual address of the zero'd cache line.

DEBUG_EVENT

Hardware debug support event. Fields:

EVENT enum
: Description of event

VALUE unsigned int
: data value

DWT_MATCH

DWT comparator matches. Fields:

NUM unsigned int
: DWT comparator number

TYPE enum
: DWT comparator configuration

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

END_COMPILE

Compilation end. Fields:

END_OF_PAGE_COUNT unsigned int
: Number of basic blocks exited due to page boundary since START_COMPILE.

FETCHFAIL_COUNT unsigned int
: Number of basic blocks exited due to ifetch failure START_COMPILE.

INST_COUNT unsigned int
: Number of instructions compiled since START_COMPILE.

NONSEQ_COUNT unsigned int
: Number of basic blocks exited due to non-sequential instructions since START_COMPILE.

PAGE_STRADDLE_COUNT unsigned int
: Number of basic blocks exited due to unaligned instructions crossing page since START_COMPILE.

EXCEPTION

Exceptions which are taken. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: The instruction set of the processor when the exception occurred.

LR unsigned int
: The value assigned to the link register.

PC unsigned int
: The location where the exception occurred.

TARGET_ISET enum
: The instruction set of the exception handler code.

TARGET_PC unsigned int
: The address the exception branches to.

VECTOR enum
: The exception vector.

EXCEPTION_END

Every exception completed.

EXCEPTION_ENTRY

Event marking the entry of an exception. All pushes of register to the stack will follow this event. Fields:

PC unsigned int
: The location where the exception occurred.

VECTOR enum
: The exception that occurred.

EXCEPTION_RAISE

Every exception raised.

EXCEPTION_RETURN

Marks the end of an exception, but on an M core does not indicate a program flow change (branch). In most situations it will be followed by an indirect branch. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ISET enum
: The instructions set of the branch instruction.

PC unsigned int
: The address of the branch instruction.

EXCEPTION_RETURN_PREBRANCH

Exception return event that must occur before PC and processor state have updated.

EXCEPTION_START

Every exception started.

EXCEPTION_VECTOR_FETCH

I-side vector fetch for M-class exception. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Result of fetch if OK

RESPONSE enum
: 0=Aborted, 1=OK

VADDR unsigned int
: Where the vector is fetched from (including VTOR)

FP_STATE

Floating point state. Fields:

FPSR unsigned int
: FPSR register value.

MASK unsigned int
: The mask indicating updated cumulative bits.

HLT

HLT instruction occurred. Fields:

IMM16 unsigned int
: The 16-bit immediate value encoded in HLT instruction.

INFO_EXCEPTION_REASON

Provide information of real cause of exception. Fields:

FaultCause enum
: Which bit of which V7M/V8M-mainline FSR/HFSR would have been set.

PC unsigned int
: The location where the exception occurred.

PHASE enum
: What we are doing with the exception.

REASONS enum
: Why we are doing it.

VECTOR enum
: The exception that occurred.

INFO_STACKING

Provide information of real cause of exception. Fields:

PHASE enum
: Are we starting or ending a sequence of stack writes.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

REASONS enum
: Why we are doing it.

INST

Every instruction executed. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DEBUG_STATE bool
: The instruction is executed in debug state.

DISASS string
: Disassembly of instruction.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The current instruction set.

ITSTATE unsigned int
: The current ITSTATE.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

MODE enum
: The mode the core is in.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

OPCODE unsigned int
: The opcode of the instruction.

PADDR unsigned int
: The physical address of the instruction.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

INST_END

Every instruction completed.

INST_START

Every instruction started. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

INST_STRADDLE

Instruction straddles boundary. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

ISB

ISB instruction occurred.

LOAD_MULTIPLE_REGS_LIST

List of destination registers (not source) of a multiple registers load instruction. Fields:

REG string
: Register to add into the list

LOCAL_MONITOR

Local monitor activity. Fields:

PADDR unsigned int
: Local Monitor Address

State enum
: State of the monitor (Open/Exclusive)

LOCKUP_CYCLE

This event is triggered if the core enters or stays in the lock-up state. Fields:

CAUSE enum
: What causes the lockup?.

LOCKUP_ENTRY

This event is triggered if the core enters the lock-up state. Fields:

PC unsigned int
: The PC when the lockup state is entered.

LOCKUP_EXIT

This event is triggered if the core leaves the lock-up state.

MODE_CHANGE

Mode change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: The new mode.

OLD_MODE enum
: The old mode.

MPU_TRANS

Address translation information. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

EXEC_PERM enum
: Execution Permission

INNERCACHE_RA bool
: Is the inner cache allocate on read

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

MEMTYPE enum
: Memory type.

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

PADDR unsigned int
: Address of the access.

PAGESIZE unsigned int
: Page size as log2(size).

READ_PERM enum
: Read Permission

REG_NUM signed int
: MPU region-number 0-15 (else 0xFF=none, 0xBB/0xDD=background/default)

SH enum
: Shareability

SIDE enum
: Inst / Data.

WRITE_PERM enum
: Write Permission

PERIODIC

Called for every quantum. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

PC unsigned int
: The address of the next instruction to be executed on this CPU.

PPB_READ_ACCESS

Trace reads from the core to memory mapped registers in the PPB (private peripheral bus) address range. Fields:

ACCESS_SIZE unsigned int
: Logarithm of the access size: 0=byte, 1=halfword, 2=word, 3=doubleword, ...

BURST_LEN unsigned int
: The number of data transfers (beats) in this burst.

DATA unsigned int
: The data transferred.

INNER_ATTRIBUTES unsigned int
: The inner attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

INST bool
: Is an instruction fetch.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOCK enum
: Normal / Locked / Exclusive.

MASTER_ID unsigned int
: The AXI master ID.

NS bool
: Is a nonsecure access.

OUTER_ATTRIBUTES unsigned int
: The outer attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

PADDR unsigned int
: Physical address of access.

PRIV bool
: Is privileged access.

RESPONSE enum
: Whether the transaction was successful, or an error occurred.

USER_FLAGS unsigned int
: Core specific additional signals.

PPB_WRITE_ACCESS

Trace writes from the core to memory mapped registers in the PPB (private peripheral bus) address range. Fields:

ACCESS_SIZE unsigned int
: Logarithm of the access size: 0=byte, 1=halfword, 2=word, 3=doubleword, ...

BURST_LEN unsigned int
: The number of data transfers (beats) in this burst.

DATA unsigned int
: The data transferred.

INNER_ATTRIBUTES unsigned int
: The inner attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

INST bool
: Is an instruction fetch.

LOCK enum
: Normal / Locked / Exclusive.

MASTER_ID unsigned int
: The AXI master ID.

NS bool
: Is a nonsecure access.

OUTER_ATTRIBUTES unsigned int
: The outer attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

PADDR unsigned int
: Physical address of access.

PRIV bool
: Is privileged access.

RESPONSE enum
: Whether the transaction was successful, or an error occurred.

USER_FLAGS unsigned int
: Core specific additional signals.

PRELOAD_DATA

Data preload from PLD instruction. Fields:

VADDR unsigned int
: Virtual address of the data that should be preloaded.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PRELOAD_INST

Instruction preload from PLI instruction. Fields:

VADDR unsigned int
: Virtual address of the instruction that should be preloaded.

PRE_CORE_LOAD

Trace just before a core load. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_CORE_STORE

Trace just before a core store. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PSTATE_PACTIVE_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

PSTATE_PACTIVE_OPMODE_TRANSITION_DENIED

Trace the denied pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_POWER_TRANSITION

Trace pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

PSTATE_PACTIVE_POWER_TRANSITION_DENIED

Trace the denied pactive and power mode transitions on PChannel. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_THREAD_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions for threads on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

RUN_STATE

Run state transition. Fields:

INST_COUNT unsigned int
: Ticks count at point of transition.

NEW enum
: New run state.

OLD enum
: Old run state.

SEMIHOSTING_CALL

Call of a semihost function occurred. Fields:

PC unsigned int
: The program counter after the semihosting call.

SEMIHOSTING_PRECALL

About to call semihost. Fields:

EL2 bool
: Use EL2 translation regime.

NS bool
: Is Non-Secure.

REG_WIDTH unsigned int
: The current register width in bytes.

SIGNAL

External signal state change. Fields:

SIGNAL enum
: Signal that changed

STATE bool
: Signal asserted state

START_COMPILE

Compilation started. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VADDR unsigned int
: Instruction where compilation begins.

STG_MULTIPLE_MICRO_OPS_ADDRESS_RANGE

Address range for MTE stores with multiple uops. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

STORE_MULTIPLE_REGS_ADDRESS_RANGE

Address range for stores with multiple registers. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

SUBOPTIMAL_LDST_RETIRED

suboptimal load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1: half-word, 2: word ...)

OPTIMAL_ALIGN_SELECTOR unsigned int
: Selector to decide the optimal alignment size (1: access width, 2: 4-bytes, 3: 8-bytes, ..., 12: 4-kbytes)

VADDR unsigned int
: The virtual address of the access

SYNC

Called for every synchronization, that is every quantum break. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

LOCAL_QUANTUM unsigned int
: The local quantum of this CPU.

LOCAL_TIME unsigned int
: The local time of this CPU.

SYNC_REASON enum
: Reason for sync.

SYSCALL

System call instruction executed. Fields:

IMM unsigned int
: Immediate value of the system call instruction.

TYPE enum
: System call type.

VADDR unsigned int
: Instruction that caused the system call.

SYSREG_READ

Trace reads from system registers. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

REG_NAME enum
: Register number

VALUE unsigned int
: The value read.

SYSREG_UPDATE32

Triggers when the system updates a register. Fields:

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

SYSREG_UPDATE64

Triggers when the system updates a register. Fields:

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

SYSREG_WRITE

Trace write to system registers. Fields:

REG_ACCESS enum
: Register access status

REG_NAME enum
: Register number

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: Value written to the register.

UNALIGNED_LDST_RETIRED

Processor unaligned load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1=>half-word, 2->word ...)

NUMBER_OF_BEATS unsigned int
: Number of accesses (beats) in this burst (Total size in bytes is ACCESS_SIZE * NUMBER_OF_BEATS)

VADDR unsigned int
: The virtual address of the access

WAYPOINT

Signals end of basic block execution. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: Origin instructions set

IS_COND bool
: Indicates if this is a conditional waypoint.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PC unsigned int
: Origin address (or 0 if unavailable)

TAKEN bool
: Indicates if this waypoint was taken.

TARGET unsigned int
: Destination address

TARGET_ISET enum
: Destination instructions set

WFE_END

WFE ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFE.

WFE_EVENT_REGISTER

WFE event register status: set/clear, reason. Fields:

INST_COUNT unsigned int
: Ticks count.

REASON enum
: Reason for set/clear. Only REASON==0 (Cleared by WFE) clears the bit, all other reasons set it.

WFE_IGNORED

WFE ignored. Fields:

EVENT bool
: This WFE was ignored because the event register was set.

INST_COUNT unsigned int
: Ticks count when ignoring WFE.

TRAPPED bool
: This WFE was trapped.

WFE_START

WFE entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFE.

WFI_END

WFI ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFI.

WFI_IGNORED

WFI ignored. Fields:

DISABLED bool
: This WFI was ignored because WFI is disabled.

INST_COUNT unsigned int
: Ticks count when ignoring WFI.

REASON enum
: specifies reason why WFI trace was ignored

TRAPPED bool
: This WFI was trapped.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

WFI_START

WFI entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFI.

WFI_WAKEUP

WFI wakeup. Fields:

INST_COUNT unsigned int
: Ticks count when WFI wakeup occurred.

REASON enum
: Reason for wakeup.

XPSR

Changes to the xPSR register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

OLD_VALUE unsigned int
: The old xPSR value

UNKNOWN unsigned int
: Bits within the register that have unknown value.

VALUE unsigned int
: The new xPSR value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.31 ARM_Cortex-M23 - trace

This section describes the trace sources.

ASYNC_MEMORY_FAULT

Context ID Register write. Fields:

FAULT unsigned int
: Fault status

PADDR unsigned int
: Physical Address (or 0 if unavailable)

ATOMIC_END_ACCESS

Bus trace access for atomics. Fields:

ACCESS_FAIL bool
: Memory access failed

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

LOAD_VALUE unsigned int
: Loaded values

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ATOMIC_START_ACCESS

Bus trace access for atomics. Fields:

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ArchMsg.Error.exit_code

Fields:

EXITCODE signed int
: Exit code to be returned

KIND string
: Component kind that invoked the exit code trace

REASON string
: Reason why the exit code trace was invoked (optional)

ArchMsg.Warning.TarmacReconstructor

Unexpected event happened during Tarmac reconstruction. Fields:

REASON string
: Reason

ArchMsg.Warning.Unpredictably Indexed PM Event Register

Fields:

IsDirect unsigned int
: Direct Access rather than indirect

N unsigned int
: Number of Accessible Registers

SEL unsigned int
: Selector

ArchMsg.Warning.branch_to_unaligned_address

Fields:

ADDR unsigned int
: Unaligned branch address

ArchMsg.Warning.dap_csw_bad_size

A write to CM3DAP CSW has an invalid size field. Fields:

DATA unsigned int
: bits[2:0] are size

ArchMsg.Warning.dcimvac_matches_watchpoint

Watchpoints matching an AArch32 DCIMVA are implementation defined.

ArchMsg.Warning.decode_fieldmismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_fracbitsnegative

VCVT Instruction is unpredictable with a negative number of fraction bits.

ArchMsg.Warning.decode_initblock

Instruction is unpredictable in an IT block.

ArchMsg.Warning.decode_initblocknotlast

Instruction is unpredictable in an IT block when not the last.

ArchMsg.Warning.decode_invalid_condition

Instruction is unpredictable when the condition is not AL.

ArchMsg.Warning.decode_invalidfieldcombination

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_invalidvalue

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_mem_hint_unallocated

There is no memory hint allocated to this bit pattern.

ArchMsg.Warning.decode_registermatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registermismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registeroutofrange

Float point instruction is unpredictable indexing beyond the end of the register bank.

ArchMsg.Warning.decode_sbzsbo

Reserved bits in the instruction are not canonical.

ArchMsg.Warning.decode_transactiontoobig

Attempt to perform a memory transaction of more than 128 bytes.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_unpred_other

Instruction is unpredictable.

ArchMsg.Warning.decode_unpreduseofpc

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13inreglist

Instruction is unpredictable with R13 in register-list.

ArchMsg.Warning.decode_useofd16orhigher

Attempt to use register D16 or higher where it is not allowed.

ArchMsg.Warning.decode_writebackandbaseinlist

Instruction is unpredictable with write back when the base register is in the transfer list.

ArchMsg.Warning.decode_zeroregistersinlist

Instruction is unpredictable when no registers are to be transfered.

ArchMsg.Warning.recursive_branch

An instruction is performing a branch that targets the same instruction. If this was intended then a WFI might be more effective. Fields:

PC unsigned int
: Address of instruction causing the branch

ArchMsg.Warning.recursive_exception

An instruction has generated an exception that targets the same instruction. Fields:

CPSR unsigned int
: Processor state

PC unsigned int
: Address of instruction causing the exception

TYPE enum
: Type of exception

ArchMsg.Warning.reentrant_vector_catch

Fields:

ADDR unsigned int
: vector address

ArchMsg.Warning.reserved_it_state

Fields:

ITSTATE unsigned int
: reserved IT state

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.secure_vector_fetch_from_nonsecure

Secure vector table is being fetched from Non-secure memory .

ArchMsg.Warning.sg_in_it_block

Arm recommends that software does not place SG (security gate instruction) inside an IT instruction block.

ArchMsg.Warning.unpredictable_a32_breakpoint

Fields:

ADDR1 unsigned int
: Address

ADDR2 unsigned int
: Breakpoint address

BAS unsigned int
: BAS Field

IS_ADDRESS_MISMATCH_BREAKPOINT bool
: Is Mismatch breakpoint, not a match breakpoint

N unsigned int
: Breakpoint number

OFFSET unsigned int
: Indicates which halfword actually considering into the A32 instruction

ArchMsg.Warning.unpredictable_banked_register_access

Fields:

R unsigned int
: R bit

SYSM unsigned int
: SYSm

ArchMsg.Warning.unpredictable_ccfail_undef

Fields:

OPCODE unsigned int
: opcode of the instruction

VADDR unsigned int
: Virtual address of the instruction being executed

ArchMsg.Warning.unpredictable_ici_bits

ICI bits do not apply or are architecturally unpredictable for instruction.

ArchMsg.Warning.unpredictable_t32_breakpoint

Fields:

ADDR1 unsigned int
: Address

ADDR2 unsigned int
: Breakpoint address

BAS unsigned int
: BAS Field

IS_ADDRESS_MISMATCH_BREAKPOINT bool
: Is Mismatch breakpoint, not a match breakpoint

N unsigned int
: Breakpoint number

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_t32_vector_catch

Fields:

ADDR unsigned int
Address
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_watchpoint_far

Fields:

LOWER_BOUND unsigned int
the lowest address accessed by the instruction that triggered the watchpoint
UPPER_BOUND unsigned int
the highest watchpointed address accessed by that instruction

ArchMsg.Warning.warning_AdvSIMDExpandImmUnexpectedZero

AdvSIMDExpandImm may treat this immediate value as UNPREDICTABLE.

ArchMsg.Warning.warning_ConditionalSMC

SMC instruction has UNPREDICTABLE effects when conditional when combined with traps.

ArchMsg.Warning.warning_access_crosses_page_boundary

Access crosses page boundary.

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault_on_both_pages

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_spanning_different_memory

Fields:

ADDR unsigned int
address of access crossing page
MEMTYPE_PAGE1 signed int
First Page's memory type (0-Normal, 1-Non-Normal)
MEMTYPE_PAGE2 signed int
Second Page's memory type (0-Normal, 1-Non-Normal)

ArchMsg.Warning.warning_access_wraps_around_memory

Fields:

WIDTH unsigned int
Address width

ArchMsg.Warning.warning_bx_from_thumbee

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR unsigned int
Destination address

ArchMsg.Warning.warning_ccsidr_unimplemented_level

Fields:

CSSELR unsigned int
current effective value of CSSELR

ArchMsg.Warning.warning_change_to_ns_when_tge_set

Attempting to change NS to 1 when HCR.TGE = 1.

ArchMsg.Warning.warning_contiguous_bit_check_abort

Fields:

ENTRY_ADDR unsigned int
address of conflicting TLB entry

ArchMsg.Warning.warning_contiguous_bit_error

Fields:

CONTIG_ADDR unsigned int
address of the TLB first read, with which this entry is expected to be contiguous
CONTIG_DATA unsigned int
expected contents based on the entry at CONTIG_ADDR
ENTRY_ADDR unsigned int
address of conflicting TLB entry
ENTRY_DATA unsigned int
contents of conflicting TLB entry

ArchMsg.Warning.warning_cp10_cp11_mismatch

Fields:

EL unsigned int
EL owning the control

ArchMsg.Warning.warning_cp10_cp11_reserved_value

Fields:

CP10 unsigned int
CP10 access
CP11 unsigned int
CP11 access

ArchMsg.Warning.warning_csselr_level_out_of_range

Fields:

IMPLEMENTED unsigned int
number of cache levels implemented
LEVEL unsigned int
written value of CSSELR.Level
TYPE enum
The type of cache selected by CSSELR.InD and TnD

ArchMsg.Warning.warning_debug_flow_control_bits_not_obeyed

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ERROR enum
: type of error resulting

REGISTER enum
: register under consideration

ArchMsg.Warning.warning_debug_register_access_during_reset

Fields:

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

ArchMsg.Warning.warning_decode_cps_inconsistent_fields

The mode setting fields in the CPS instruction are inconsistent.

ArchMsg.Warning.warning_decode_invalid_state

Instruction is unpredictable in current exception-level/security state.

ArchMsg.Warning.warning_default_cacheable_mmu_on

Default cacheable is enabled (HCR.DC = 1) while MMU is enabled.

ArchMsg.Warning.warning_default_cacheable_vmmu_off

Default cacheable is enabled (HCR.DC = 1) while HCR.VM = 0.

ArchMsg.Warning.warning_deprecated_wvr_bit_2_set

Fields:

N unsigned int
: index

WVR unsigned int
: DBGWVR

ArchMsg.Warning.warning_eret_while_software_step_active_pending

Missing ISB between setting MDSCR_EL1.SS and ERET.

ArchMsg.Warning.warning_exclusive_to_non_normal

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_exclusive_to_non_writeback

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_execute_from_device_memory

An attempt was made to execute from Device memory.

ArchMsg.Warning.warning_illegal_cpsr_mode

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NEW_MODE unsigned int
New value of CPSR.M

ArchMsg.Warning.warning_illegal_srs_mode

Fields:

MODE unsigned int
mode for Banked SP

ArchMsg.Warning.warning_implementation_defined_read_debug_register_in_SCS

Read Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

PPB_OFFSET unsigned int
debug register offset

ArchMsg.Warning.warning_implementation_defined_sequential_security_transitions_supported

The behaviour of a sequential instruction fetches that cross from non-secure to secure memory and contain SG instruction is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_implementation_defined_stack_limit_check_supported

It is IMPLEMENTATION DEFINED whether stack pointer limit checking is performed for the this instructions. Fields:

ADDRESS unsigned int
Address of instruction
OPCODE unsigned int
instruction opcode

ArchMsg.Warning.warning_implementation_defined_write_debug_register_in_SCS

Write Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

DATA unsigned int
data attempted to be written
PPB_OFFSET unsigned int
debug register offset

ArchMsg.Warning.warning_invalid_tcr_granule

Fields:

REQUEST signed int
page size requested (or 0 for reserved)
SUBSTITUTE signed int
best guess available page size
TG_ID bool
bits TG0 or TG1

ArchMsg.Warning.warning_load_multiple_user_registers_from_user_mode

An LDM(user registers) instruction executed from user mode.

ArchMsg.Warning.warning_load_pc_from_unaligned

PC loaded from an unaligned location.

ArchMsg.Warning.warning_mte_in_mem_attr_not_supported

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ATTR unsigned int
Memory attribute

ArchMsg.Warning.warning_reserved_breakpoint_state_match

Fields:

IS_BREAKPOINT unsigned int
Is breakpoint, not watchpoint

ArchMsg.Warning.warning_shareability

Unpredictable: combination of the 1st and 2nd stages of translation is Normal Inner Non-Cacheable, Outer Non-Cacheable.

ArchMsg.Warning.warning_software_step_set_while_enabled

MDSCR_EL1.SS set to 1 while software step debug exceptions are enabled.

ArchMsg.Warning.warning_thumb_instruction_wraps_around_memory

Thumb instruction warps around memory.

ArchMsg.Warning.warning_ttbr_sbz_bits_are_not_zero

Fields:

MSB unsigned int
x-1
TTBR enum
which TTBR register

ArchMsg.Warning.warning_unaligned_address_dbgdtrrx_write

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_address_dbgdtrtx_read

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_device

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_strongly_ordered

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unknown_sau_rnr

SAU_RNR was set to an unsupported value.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_AIRCR_PRIP_and_BFHFNMINP

The effect of setting both AIRCR.BFHFNMINP and AIRCR.PRIP to 1 is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTCLRACTIVE_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTCLRACTIVE if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_and_SYSRESETREQ

When the processor is halted in Debug state, if a write to the register writes a 1 to both VECTRESET and SYSRESETREQ, the behavior is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTRESET if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_incorrect_VKEY

The value 0x05FA must be written to AIRCR.VKEY, otherwise the register write is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_EXE_RETURN_Reserved_Bit

EXC_RETURN[23:7] are reserved with the special condition that all bits should be written as one. Values other than all 1s are UNPREDICTABLE. Fields:

EXC_RETURN `unsigned int`
: Special PC value

SBO_MASK `unsigned int`
: Which bits Should Be One to not UNPRED

ArchMsg.Warning.warning_unpredictable_change_to_DEMCR_MON_STEP_at_insufficient_priority

The effect of changing DEMCR.MON_STEP at an execution priority that is lower than the priority of the DebugMonitor exception is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_change_to_priority_of_active_exception

Changing the priority of an active exception is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_clear_lspact

Fields:

DRVEXC_TAKEN `unsigned int`
: derived exception taken not pended

REACHED_LAST_STORE `unsigned int`
: only the last faulted

ArchMsg.Warning.warning_unpredictable_dbg_exit_unaligned_dlr

Debug exit to AArch32 with DLR[0] set to 1. PC[0] can be set to 0 or to DLR[0].

ArchMsg.Warning.warning_unpredictable_exception_catch

Generation of an exception catch event is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_inconsistent_state

state on exception return is unpredictable.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_exception_return_instruction

exception return instruction is unpredictable.

ArchMsg.Warning.warning_unpredictable_in_debug_state

Instruction is UNPREDICTABLE when executed in debug state.

ArchMsg.Warning.warning_unpredictable_mair_encoding

Fields:

ATTR unsigned int
: Memory attribute

ArchMsg.Warning.warning_unpredictable_pmu_counter_access

Access to PMU counters from non-secure EL0 or EL1 is UNPREDICTABLE with MDCR_EL2.HPMN set to 0.

ArchMsg.Warning.warning_unpredictable_prioritization_breakpoint_match_vector_catch

Two events with the same priority occurred at the same instruction: Address Matching Vector Catch debug event, and Breakpoint debug event It is constrained unpredictable which is taken.

ArchMsg.Warning.warning_unpredictable_stack_selection

UNPREDICTABLE state on context switch.

ArchMsg.Warning.warning_unpredictable_tsize_out_of_range

Fields:

TCR enum
: Which TCR

TSIZE signed int
: Value of TSize

TSIZE_MAX signed int
: The maximum allowed value

TTBR unsigned int
: Which TTBR region

ArchMsg.Warning.warning_unpredictable_unaligned_pc_as_base_register

The PC value must be word-aligned, otherwise the behavior of the instruction is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_unaligned_pop_stack

Fields:

FRAME_ADDR unsigned int
: unaligned stack frame address

ArchMsg.Warning.warning_unpredictable_vmsa_memattrib

Fields:

TEXCB unsigned int
: texcb value

TEX_REMAP bool
: use tex remap

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_vtcr_t0sz_sl0

Fields:

SL0 signed int
: VTCR.SL0, starting level for stage 2 translation table walks

T0SZ signed int
: VTCR.T0SZ, required input address range

ArchMsg.Warning.warning_unpredictable_write_DHCSR

Fields:

NEW unsigned int
: data attempted to be written

OLD unsigned int
: previous value

ArchMsg.Warning.warning_unpredictable_write_SHCSR

UNPREDICTABLE setting a pending bit to 1 for an exception with priority >= execution priority.

ArchMsg.Warning.warning_unsupported_access_to_memory_mapped_register

Access to memory mapped register is unsupported.

ArchMsg.Warning.warning_user_jmcr_access

Fields:

WRITE unsigned int
: access is write

ArchMsg.Warning.warning_wcr_mask_and_bas

DBGWCR_EL1.MASK is non-zero and BAS is not 0xff.

ArchMsg.Warning.warning_wcr_mask_reserved

DBGWCR_ELn.MASK is set to a reserved value.

ArchMsg.Warning.warning_wcr_non_configuous_bas

DBGWCR_EL1.BAS has non-contiguous set of ones.

ArchMsg.Warning.warning_wvr_masked_bit_not_zero

DBGWVRn_EL1.VA contains a bit masked by DBGWCRn_EL1.MASK that is not zero.

ArchMsg.warning_unpred_system_register_access

Fields:

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

BRANCH_MISPREDICT

Simulating branch mispredict. Fields:

PC unsigned int
: Origin address (or 0 if unavailable)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

BRA_DIR

Direct branches, to immediate address. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR

Indirect branches, perhaps to a register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR_CCFAIL

Indirect branches fail, perhaps to a register.

BRA_RET_CCFAIL

Branche return fail, perhaps to a register.

CCFAIL

Conditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CCFAIL_UNC

Unconditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS

Conditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS_UNC

Unconditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CODE_CACHE_MAINT

Code cache maintenance. Fields:

END_INCL unsigned int
: Inclusive end addres for by VA or PA requests

NS bool
: Non-secure world for PA

START unsigned int
: Start address for by VA or PA requests

TYPE enum
: Request type.

COMPILE_BLOCK_END

Last instruction of basic block translated. Fields:

VADDR unsigned int
: Address of next instruction after this basic block

COMPILE_BLOCK_START

First instruction of basic block translated. Fields:

VADDR unsigned int
: Address of first instruction in basic block

COMPILE_INST

ARM instruction compiled. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DISASS string
: Disassembly of the instruction.

ISET enum
: The instruction set of this instruction.

ITSTATE unsigned int
: The ITSTATE current for the instruction.

OPCODE unsigned int
: The opcode of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

CONTEXTIDR

Context ID Register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

NS bool
: Secure or nonsecure banked register is accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

CONTEXT_SYNC

Called for every context synchronization event. Fields:

ARCH_CSE bool
: Architectural Context Synchronization Event

SYSREG_SYNC bool
: System register synchronization event

CORE_ENDIAN

Core BE8 Big-Endian state changed. Fields:

BE8 bool
: Core BE8 Big-Endian state

CORE_INFO

Static processor attributes. Only triggered by a call to DumpState(). Fields:

ARCH_PROFILE enum
: The architecture profile of the core.

CLUSTER_ID unsigned int
: The cluster ID of this processor.

CORE_NUM unsigned int
: The number of this core in an MP processor.

DETAILED_CORE_FEATURES enum
: Detailed core features as a bitfield. Enum values are bitmasks to extract the info from the field

FPU_VERSION enum
: The VFP version implemented by the core.

MEM_ARCH enum
: The memory architecture of the core.

MIDR_EL1 unsigned int
: MIDR_EL1 value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NUM_CORES unsigned int
: The number of cores in this MP processor.

QUANTUM_SIZE unsigned int
: The default quantum size of the core.

SECURITY_FEATURES bool
: Does the core have security features?

CORE_LOADS

Processor load accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CORE_REGS

Changes of the core registers R0 to R14. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ID unsigned int
: The register number, 0 to 14.

OLD_VALUE unsigned int
: The old value overwritten.

PHYS_ID enum
: The physical register accessed.

VALUE unsigned int
: The new value written to the register.

CORE_STORES

Processor store accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CRYPTO_SPEC

Every crypto instruction speculatively executed.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DATA_CACHE_SET_ALLOC_TAGS

Set allocation tags from DC GVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of memory in bytes.

TAG unsigned int
: The value of the tag being set.

VADDR unsigned int
: Virtual address used for setting allocation tags.

DATA_CACHE_ZERO

Zero and invalidate cache line from DC ZVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of zero'd memory in bytes.

VADDR unsigned int
: Virtual address of the zero'd cache line.

DEBUG_EVENT

Hardware debug support event. Fields:

EVENT enum
: Description of event

VALUE unsigned int
: data value

DWT_MATCH

DWT comparator matches. Fields:

NUM unsigned int
: DWT comparator number

TYPE enum
: DWT comparator configuration

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

END_COMPILE

Compilation end. Fields:

END_OF_PAGE_COUNT unsigned int
: Number of basic blocks exited due to page boundary since START_COMPILE.

FETCHFAIL_COUNT unsigned int
: Number of basic blocks exited due to ifetch failure START_COMPILE.

INST_COUNT unsigned int
: Number of instructions compiled since START_COMPILE.

NONSEQ_COUNT unsigned int
: Number of basic blocks exited due to non-sequential instructions since START_COMPILE.

PAGE_STRADDLE_COUNT unsigned int
: Number of basic blocks exited due to unaligned instructions crossing page since START_COMPILE.

EXCEPTION

Exceptions which are taken. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: The instruction set of the processor when the exception occurred.

LR unsigned int
: The value assigned to the link register.

NS enum
: The core's non-secure bit.

PC unsigned int
: The location where the exception occurred.

TARGET_ISET enum
: The instruction set of the exception handler code.

TARGET_PC unsigned int
: The address the exception branches to.

VECTOR enum
: The exception vector.

EXCEPTION_END

Every exception completed. Fields:

NS enum
: The core's non-secure bit.

EXCEPTION_ENTRY

Event marking the entry of an exception. All pushes of register to the stack will follow this event. Fields:

PC unsigned int
: The location where the exception occurred.

VECTOR enum
: The exception that occurred.

EXCEPTION_RAISE

Every exception raised.

EXCEPTION_RETURN

Marks the end of an exception, but on an M core does not indicate a program flow change (branch). In most situations it will be followed by an indirect branch. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

PC unsigned int
: The address of the branch instruction.

EXCEPTION_RETURN_PREBRANCH

Exception return event that must occur before PC and processor state have updated.

EXCEPTION_START

Every exception started. Fields:

NS enum
: The core's non-secure bit.

EXCEPTION_VECTOR_FETCH

I-side vector fetch for M-class exception. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Result of fetch if OK

RESPONSE enum
: 0=Aborted, 1=OK

VADDR unsigned int
: Where the vector is fetched from (including VTOR)

FP_STATE

Floating point state. Fields:

FPSR unsigned int
: FPSR register value.

MASK unsigned int
: The mask indicating updated cumulative bits.

HLT

HLT instruction occurred. Fields:

IMM16 unsigned int
: The 16-bit immediate value encoded in HLT instruction.

INFO_EXCEPTION_REASON

Provide information of real cause of exception. Fields:

FaultCause enum
: Which bit of which V7M/V8M-mainline FSR/HFSR would have been set.

PC unsigned int
: The location where the exception occurred.

PHASE enum
: What we are doing with the exception.

REASONS enum
: Why we are doing it.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SecurityState enum
: Privacy State of exception

VECTOR enum
: The exception that occurred.

INFO_STACKING

Provide information of real cause of exception. Fields:

PHASE enum
: Are we starting or ending a sequence of stack writes.

REASONS enum
: Why we are doing it.

INST

Every instruction executed. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DEBUG_STATE bool
: The instruction is executed in debug state.

DISASS string
: Disassembly of instruction.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The current instruction set.

ITSTATE unsigned int
: The current ITSTATE.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

MODE enum
: The mode the core is in.

NS unsigned int
: The core's non-secure bit.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

OPCODE unsigned int
: The opcode of the instruction.

PADDR unsigned int
: The physical address of the instruction.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

INST_END

Every instruction completed.

INST_START

Every instruction started. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

INST_STRADDLE

Instruction straddles boundary. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

ISB

ISB instruction occurred.

LOAD_MULTIPLE_REGS_LIST

List of destination registers (not source) of a multiple registers load instruction. Fields:

REG string
: Register to add into the list

LOCAL_MONITOR

Local monitor activity. Fields:

PADDR unsigned int
: Local Monitor Address

State enum
: State of the monitor (Open/Exclusive)

LOCKUP_CYCLE

This event is triggered if the core enters or stays in the lock-up state. Fields:

CAUSE enum
: What causes the lockup?.

LOCKUP_ENTRY

This event is triggered if the core enters the lock-up state. Fields:

PC unsigned int
: The PC when the lockup state is entered.

LOCKUP_EXIT

This event is triggered if the core leaves the lock-up state.

MODE_CHANGE

Mode change. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: The new mode.

OLD_MODE enum
: The old mode.

MPU_TRANS

Address translation information. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

EXEC_PERM enum
: Execution Permission

IDAU_REG_NUM enum
: IDAU region-number or code

INNERCACHE_RA bool
: Is the inner cache allocate on read

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

MEMTYPE enum
: Memory type.

NSDESC enum
: Security-attributes of the transaction.

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

PADDR unsigned int
: Address of the access.

PAGESIZE unsigned int
: Page size as log2(size).

READ_PERM enum
: Read Permission

REG_NUM signed int
: MPU region-number 0-15 (else 0xFF=none, 0xBB/0xDD=background/default)

SAU_REG_NUM enum
: SAU region-number or code

SAU_RGN_TYPE enum
: Privacy Region Type

SH enum
: Shareability

SIDE enum
: Inst / Data.

WRITE_PERM enum
: Write Permission

PERIODIC

Called for every quantum. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: The instruction count of this CPU.

PC unsigned int
: The address of the next instruction to be executed on this CPU.

PPB_READ_ACCESS

Trace reads from the core to memory mapped registers in the PPB (private peripheral bus) address range. Fields:

ACCESS_SIZE unsigned int
: Logarithm of the access size: 0=byte, 1=halfword, 2=word, 3=doubleword, ...

BURST_LEN unsigned int
: The number of data transfers (beats) in this burst.

DATA unsigned int
: The data transferred.

INNER_ATTRIBUTES unsigned int
: The inner attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

INST bool
: Is an instruction fetch.

LOCK enum
: Normal / Locked / Exclusive.

MASTER_ID unsigned int
: The AXI master ID.

NS bool
: Is a nonsecure access.

OUTER_ATTRIBUTES unsigned int
: The outer attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

PADDR unsigned int
: Physical address of access.

PRIV bool
: Is privileged access.

RESPONSE enum
: Whether the transaction was successful, or an error occurred.

USER_FLAGS unsigned int
: Core specific additional signals.

PPB_WRITE_ACCESS

Trace writes from the core to memory mapped registers in the PPB (private peripheral bus) address range. Fields:

ACCESS_SIZE unsigned int
: Logarithm of the access size: 0=byte, 1=halfword, 2=word, 3=doubleword, ...

BURST_LEN unsigned int
: The number of data transfers (beats) in this burst.

DATA unsigned int
: The data transferred.

INNER_ATTRIBUTES unsigned int
: The inner attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

INST bool
: Is an instruction fetch.

LOCK enum
: Normal / Locked / Exclusive.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MASTER_ID unsigned int
: The AXI master ID.

NS bool
: Is a nonsecure access.

OUTER_ATTRIBUTES unsigned int
: The outer attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

PADDR unsigned int
: Physical address of access.

PRIV bool
: Is privileged access.

RESPONSE enum
: Whether the transaction was successful, or an error occurred.

USER_FLAGS unsigned int
: Core specific additional signals.

PRELOAD_DATA

Data preload from PLD instruction. Fields:

VADDR unsigned int
: Virtual address of the data that should be preloaded.

PRELOAD_INST

Instruction preload from PLI instruction. Fields:

VADDR unsigned int
: Virtual address of the instruction that should be preloaded.

PRE_CORE_LOAD

Trace just before a core load. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_CORE_STORE

Trace just before a core store. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PSTATE_PACTIVE_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

PSTATE_PACTIVE_OPMODE_TRANSITION_DENIED

Trace the denied pactive and Opmode mode transitions on PChannel. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_POWER_TRANSITION

Trace pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

PSTATE_PACTIVE_POWER_TRANSITION_DENIED

Trace the denied pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_THREAD_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions for threads on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

RUN_STATE

Run state transition. Fields:

INST_COUNT unsigned int
: Ticks count at point of transition.

NEW enum
: New run state.

OLD enum
: Old run state.

SEMIHOSTING_CALL

Call of a semihost function occurred. Fields:

PC unsigned int
: The program counter after the semihosting call.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SEMIHOSTING_PRECALL

About to call semihost. Fields:

EL2 bool
: Use EL2 translation regime.

NS bool
: Is Non-Secure.

REG_WIDTH unsigned int
: The current register width in bytes.

SIGNAL

External signal state change. Fields:

SIGNAL enum
: Signal that changed

STATE bool
: Signal asserted state

START_COMPILE

Compilation started. Fields:

VADDR unsigned int
: Instruction where compilation begins.

STG_MULTIPLE_MICRO_OPS_ADDRESS_RANGE

Address range for MTE stores with multiple uops. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

STORE_MULTIPLE_REGS_ADDRESS_RANGE

Address range for stores with multiple registers. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

SUBOPTIMAL_LDST_RETIRED

suboptimal load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1: half-word, 2: word ...)

OPTIMAL_ALIGN_SELECTOR unsigned int
: Selector to decide the optimal alignment size (1: access width, 2: 4-bytes, 3: 8-bytes, ..., 12: 4-kbytes)

VADDR unsigned int
: The virtual address of the access

SYNC

Called for every synchronization, that is every quantum break. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: The instruction count of this CPU.

LOCAL_QUANTUM unsigned int
: The local quantum of this CPU.

LOCAL_TIME unsigned int
: The local time of this CPU.

SYNC_REASON enum
: Reason for sync.

SYSCALL

System call instruction executed. Fields:

IMM unsigned int
: Immediate value of the system call instruction.

TYPE enum
: System call type.

VADDR unsigned int
: Instruction that caused the system call.

SYSREG_READ

Trace reads from system registers. Fields:

REG_NAME enum
: Register number

VALUE unsigned int
: The value read.

SYSREG_UPDATE32

Triggers when the system updates a register. Fields:

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

SYSREG_UPDATE64

Triggers when the system updates a register. Fields:

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

SYSREG_WRITE

Trace write to system registers. Fields:

REG_ACCESS enum
: Register access status

REG_NAME enum
: Register number

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VALUE unsigned int
: Value written to the register.

UNALIGNED_LDST_RETIRED

Processor unaligned load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1=>half-word, 2->word ...)

NUMBER_OF_BEATS unsigned int
: Number of accesses (beats) in this burst (Total size in bytes is ACCESS_SIZE * NUMBER_OF_BEATS)

VADDR unsigned int
: The virtual address of the access

WAYPOINT

Signals end of basic block execution. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: Origin instructions set

IS_COND bool
: Indicates if this is a conditional waypoint.

PC unsigned int
: Origin address (or 0 if unavailable)

TAKEN bool
: Indicates if this waypoint was taken.

TARGET unsigned int
: Destination address

TARGET_ISET enum
: Destination instructions set

WFE_END

WFE ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFE.

WFE_EVENT_REGISTER

WFE event register status: set/clear, reason. Fields:

INST_COUNT unsigned int
: Ticks count.

REASON enum
: Reason for set/clear. Only REASON==0 (Cleared by WFE) clears the bit, all other reasons set it.

WFE_IGNORED

WFE ignored. Fields:

EVENT bool
: This WFE was ignored because the event register was set.

INST_COUNT unsigned int
: Ticks count when ignoring WFE.

TRAPPED bool
: This WFE was trapped.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

WFE_START

WFE entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFE.

WFI_END

WFI ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFI.

WFI_IGNORED

WFI ignored. Fields:

DISABLED bool
: This WFI was ignored because WFI is disabled.

INST_COUNT unsigned int
: Ticks count when ignoring WFI.

REASON enum
: specifies reason why WFI trace was ignored

TRAPPED bool
: This WFI was trapped.

WFI_START

WFI entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFI.

WFI_WAKEUP

WFI wakeup. Fields:

INST_COUNT unsigned int
: Ticks count when WFI wakeup occurred.

REASON enum
: Reason for wakeup.

XPSR

Changes to the xPSR register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

OLD_VALUE unsigned int
: The old xPSR value

UNKNOWN unsigned int
: Bits within the register that have unknown value.

VALUE unsigned int
: The new xPSR value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.32 ARM_Cortex-M3 - trace

This section describes the trace sources.

ASYNC_MEMORY_FAULT

Context ID Register write. Fields:

FAULT unsigned int
: Fault status

PADDR unsigned int
: Physical Address (or 0 if unavailable)

ATOMIC_END_ACCESS

Bus trace access for atomics. Fields:

ACCESS_FAIL bool
: Memory access failed

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

LOAD_VALUE unsigned int
: Loaded values

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ATOMIC_START_ACCESS

Bus trace access for atomics. Fields:

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ArchMsg.Error.exit_code

Fields:

EXITCODE signed int
: Exit code to be returned

KIND string
: Component kind that invoked the exit code trace

REASON string
: Reason why the exit code trace was invoked (optional)

ArchMsg.Warning.TarmacReconstructor

Unexpected event happened during Tarmac reconstruction. Fields:

REASON string
: Reason

ArchMsg.Warning.dap_csw_bad_size

A write to CM3DAP CSW has an invalid size field. Fields:

DATA unsigned int
: bits[2:0] are size

ArchMsg.Warning.recursive_branch

An instruction is performing a branch that targets the same instruction. If this was intended then a WFI might be more effective. Fields:

PC unsigned int
: Address of instruction causing the branch

ArchMsg.Warning.recursive_exception

An instruction has generated an exception that targets the same instruction. Fields:

CPSR unsigned int
: Processor state

PC unsigned int
: Address of instruction causing the exception

TYPE enum
: Type of exception

ArchMsg.Warning.secure_vector_fetch_from_nonsecure

Secure vector table is being fetched from Non-secure memory .

ArchMsg.Warning.sg_in_it_block

Arm recommends that software does not place SG (security gate instruction) inside an IT instruction block.

ArchMsg.warning_unpred_system_register_access

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

BRANCH_MISPREDICT

Simulating branch mispredict. Fields:

PC unsigned int
: Origin address (or 0 if unavailable)

BRA_DIR

Direct branches, to immediate address. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR

Indirect branches, perhaps to a register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR_CCFAIL

Indirect branches fail, perhaps to a register.

BRA_RET_CCFAIL

Branche return fail, perhaps to a register.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CCFAIL

Conditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCFAIL_UNC

Unconditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS

Conditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS_UNC

Unconditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CODE_CACHE_MAINT

Code cache maintenance. Fields:

END_INCL unsigned int
: Inclusive end addres for by VA or PA requests

NS bool
: Non-secure world for PA

START unsigned int
: Start address for by VA or PA requests

TYPE enum
: Request type.

COMPILE_BLOCK_END

Last instruction of basic block translated. Fields:

VADDR unsigned int
: Address of next instruction after this basic block

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

COMPILE_BLOCK_START

First instruction of basic block translated. Fields:

VADDR unsigned int
: Address of first instruction in basic block

COMPILE_INST

ARM instruction compiled. Fields:

DISASS string
: Disassembly of the instruction.

ISET enum
: The instruction set of this instruction.

ITSTATE unsigned int
: The ITSTATE current for the instruction.

OPCODE unsigned int
: The opcode of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

CONTEXTIDR

Context ID Register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

NS bool
: Secure or nonsecure banked register is accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

CONTEXT_SYNC

Called for every context synchronization event. Fields:

ARCH_CSE bool
: Architectural Context Synchronization Event

SYSREG_SYNC bool
: System register synchronization event

CORE_ENDIAN

Core BE8 Big-Endian state changed. Fields:

BE8 bool
: Core BE8 Big-Endian state

CORE_INFO

Static processor attributes. Only triggered by a call to DumpState(). Fields:

ARCH_PROFILE enum
: The architecture profile of the core.

CLUSTER_ID unsigned int
: The cluster ID of this processor.

CORE_NUM unsigned int
: The number of this core in an MP processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DETAILED_CORE_FEATURES enum
: Detailed core features as a bitfield. Enum values are bitmasks to extract the info from the field

FPU_VERSION enum
: The VFP version implemented by the core.

MEM_ARCH enum
: The memory architecture of the core.

MIDR_EL1 unsigned int
: MIDR_EL1 value

NUM_CORES unsigned int
: The number of cores in this MP processor.

QUANTUM_SIZE unsigned int
: The default quantum size of the core.

SECURITY_FEATURES bool
: Does the core have security features?

CORE_LOADS

Processor load accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VADDR unsigned int
: The virtual address of the access.

CORE_REGS

Changes of the core registers R0 to R14. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number, 0 to 14.

OLD_VALUE unsigned int
: The old value overwritten.

PHYS_ID enum
: The physical register accessed.

VALUE unsigned int
: The new value written to the register.

CORE_STORES

Processor store accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CRYPTO_SPEC

Every crypto instruction speculatively executed.

DATA_CACHE_SET_ALLOC_TAGS

Set allocation tags from DC GVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of memory in bytes.

TAG unsigned int
: The value of the tag being set.

VADDR unsigned int
: Virtual address used for setting allocation tags.

DATA_CACHE_ZERO

Zero and invalidate cache line from DC ZVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of zero'd memory in bytes.

VADDR unsigned int
: Virtual address of the zero'd cache line.

DEBUG_EVENT

Hardware debug support event. Fields:

EVENT enum
: Description of event

VALUE unsigned int
: data value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DWT_MATCH

DWT comparator matches. Fields:

NUM unsigned int
: DWT comparator number

TYPE enum
: DWT comparator configuration

END_COMPILE

Compilation end. Fields:

END_OF_PAGE_COUNT unsigned int
: Number of basic blocks exited due to page boundary since START_COMPILE.

FETCHFAIL_COUNT unsigned int
: Number of basic blocks exited due to ifetch failure START_COMPILE.

INST_COUNT unsigned int
: Number of instructions compiled since START_COMPILE.

NONSEQ_COUNT unsigned int
: Number of basic blocks exited due to non-sequential instructions since START_COMPILE.

PAGE_STRADDLE_COUNT unsigned int
: Number of basic blocks exited due to unaligned instructions crossing page since START_COMPILE.

EXCEPTION

Exceptions which are taken. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: The instruction set of the processor when the exception occurred.

LR unsigned int
: The value assigned to the link register.

PC unsigned int
: The location where the exception occurred.

TARGET_ISET enum
: The instruction set of the exception handler code.

TARGET_PC unsigned int
: The address the exception branches to.

VECTOR enum
: The exception vector.

EXCEPTION_END

Every exception completed.

EXCEPTION_ENTRY

Event marking the entry of an exception. All pushes of register to the stack will follow this event. Fields:

PC unsigned int
: The location where the exception occurred.

VECTOR enum
: The exception that occurred.

EXCEPTION_RAISE

Every exception raised.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

EXCEPTION_RETURN

Marks the end of an exception, but on an M core does not indicate a program flow change (branch). In most situations it will be followed by an indirect branch. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

PC unsigned int
: The address of the branch instruction.

EXCEPTION_RETURN_PREBRANCH

Exception return event that must occur before PC and processor state have updated.

EXCEPTION_START

Every exception started.

EXCEPTION_VECTOR_FETCH

I-side vector fetch for M-class exception. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Result of fetch if OK

RESPONSE enum
: 0=Aborted, 1=OK

VADDR unsigned int
: Where the vector is fetched from (including VTOR)

FP_STATE

Floating point state. Fields:

FPSR unsigned int
: FPSR register value.

MASK unsigned int
: The mask indicating updated cumulative bits.

HLT

HLT instruction occurred. Fields:

IMM16 unsigned int
: The 16-bit immediate value encoded in HLT instruction.

INFO_EXCEPTION_REASON

Provide information of real cause of exception. Fields:

FaultCause enum
: Which bit of which V7M/V8M-mainline FSR/HFSR would have been set.

PC unsigned int
: The location where the exception occurred.

PHASE enum
: What we are doing with the exception.

REASONS enum
: Why we are doing it.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VECTOR enum
The exception that occurred.

INFO_STACKING

Provide information of real cause of exception. Fields:

PHASE enum
Are we starting or ending a sequence of stack writes.
REASONS enum
Why we are doing it.

INST

Every instruction executed. Fields:

CORE_NUM unsigned int
Core number in a multi processor.
CURRENT_TIME unsigned int
The core's current time, as simulated time plus local time.
DEBUG_STATE bool
The instruction is executed in debug state.
DISASS string
Disassembly of instruction.
INST_COUNT unsigned int
The core's instruction counter, starting at 1 for the first instruction.
ISET enum
The current instruction set.
ITSTATE unsigned int
The current ITSTATE.
LOCAL_TIME unsigned int
The core's local time, relative to the current quantum.
MODE enum
The mode the core is in.
NSDESC unsigned int
The physical address non-secure bit.
NSDESC2 unsigned int
The second page physical address non-secure bit.
OPCODE unsigned int
The opcode of the instruction.
PADDR unsigned int
The physical address of the instruction.
PADDR2 unsigned int
If different from PADDR, the physical address of the second page of the instruction.
PC unsigned int
The address of the instruction.
SIZE unsigned int
The size of the instruction in bytes.

INST_END

Every instruction completed.

INST_START

Every instruction started. Fields:

ISET enum
The current instruction set.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

INST_STRADDLE

Instruction straddles boundary. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

ISB

ISB instruction occurred.

ITM

Instrumentation Trace Macrocell. Fields:

ITM_PACKET_TYPE enum
: ITM and DWT packets type.

PACKET_HEADER unsigned int
: ITM Packet Header.

PACKET_PAYLOAD unsigned int
: ITM Packet Payload.

LOAD_MULTIPLE_REGS_LIST

List of destination registers (not source) of a multiple registers load instruction. Fields:

REG string
: Register to add into the list

LOCAL_MONITOR

Local monitor activity. Fields:

PADDR unsigned int
: Local Monitor Address

State enum
: State of the monitor (Open/Exclusive)

LOCKUP_CYCLE

This event is triggered if the core enters or stays in the lock-up state. Fields:

CAUSE enum
: What causes the lockup?.

LOCKUP_ENTRY

This event is triggered if the core enters the lock-up state. Fields:

PC unsigned int
: The PC when the lockup state is entered.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOCKUP_EXIT

This event is triggered if the core leaves the lock-up state.

MODE_CHANGE

Mode change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: The new mode.

OLD_MODE enum
: The old mode.

MPU_TRANS

Address translation information. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

EXEC_PERM enum
: Execution Permission

INNERCACHE_RA bool
: Is the inner cache allocate on read

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

MEMTYPE enum
: Memory type.

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

PADDR unsigned int
: Address of the access.

PAGESIZE unsigned int
: Page size as log2(size).

READ_PERM enum
: Read Permission

REG_NUM signed int
: MPU region-number 0-15 (else 0xFF=none, 0xBB/0xDD=background/default)

SH enum
: Shareability

SIDE enum
: Inst / Data.

WRITE_PERM enum
: Write Permission

PERIODIC

Called for every quantum. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PC unsigned int
: The address of the next instruction to be executed on this CPU.

PPB_READ_ACCESS

Trace reads from the core to memory mapped registers in the PPB (private peripheral bus) address range. Fields:

ACCESS_SIZE unsigned int
: Logarithm of the access size: 0=byte, 1=halfword, 2=word, 3=doubleword, ...

BURST_LEN unsigned int
: The number of data transfers (beats) in this burst.

DATA unsigned int
: The data transferred.

INNER_ATTRIBUTES unsigned int
: The inner attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

INST bool
: Is an instruction fetch.

LOCK enum
: Normal / Locked / Exclusive.

MASTER_ID unsigned int
: The AXI master ID.

NS bool
: Is a nonsecure access.

OUTER_ATTRIBUTES unsigned int
: The outer attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

PADDR unsigned int
: Physical address of access.

PRIV bool
: Is privileged access.

RESPONSE enum
: Whether the transaction was successful, or an error occurred.

USER_FLAGS unsigned int
: Core specific additional signals.

PPB_WRITE_ACCESS

Trace writes from the core to memory mapped registers in the PPB (private peripheral bus) address range. Fields:

ACCESS_SIZE unsigned int
: Logarithm of the access size: 0=byte, 1=halfword, 2=word, 3=doubleword, ...

BURST_LEN unsigned int
: The number of data transfers (beats) in this burst.

DATA unsigned int
: The data transferred.

INNER_ATTRIBUTES unsigned int
: The inner attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

INST bool
: Is an instruction fetch.

LOCK enum
: Normal / Locked / Exclusive.

MASTER_ID unsigned int
: The AXI master ID.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS bool
: Is a nonsecure access.

OUTER_ATTRIBUTES unsigned int
: The outer attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

PADDR unsigned int
: Physical address of access.

PRIV bool
: Is privileged access.

RESPONSE enum
: Whether the transaction was successful, or an error occurred.

USER_FLAGS unsigned int
: Core specific additional signals.

PRELOAD_DATA

Data preload from PLD instruction. Fields:

VADDR unsigned int
: Virtual address of the data that should be preloaded.

PRELOAD_INST

Instruction preload from PLI instruction. Fields:

VADDR unsigned int
: Virtual address of the instruction that should be preloaded.

PRE_CORE_LOAD

Trace just before a core load. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_CORE_STORE

Trace just before a core store. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PSTATE_PACTIVE_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

PSTATE_PACTIVE_OPMODE_TRANSITION_DENIED

Trace the denied pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_POWER_TRANSITION

Trace pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

PSTATE_PACTIVE_POWER_TRANSITION_DENIED

Trace the denied pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_THREAD_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions for threads on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

RUN_STATE

Run state transition. Fields:

INST_COUNT unsigned int
: Ticks count at point of transition.

NEW enum
: New run state.

OLD enum
: Old run state.

SEMIHOSTING_CALL

Call of a semihost function occurred. Fields:

PC unsigned int
: The program counter after the semihosting call.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SEMIHOSTING_PRECALL

About to call semihost. Fields:

EL2 bool
: Use EL2 translation regime.

NS bool
: Is Non-Secure.

REG_WIDTH unsigned int
: The current register width in bytes.

SIGNAL

External signal state change. Fields:

SIGNAL enum
: Signal that changed

STATE bool
: Signal asserted state

START_COMPILE

Compilation started. Fields:

VADDR unsigned int
: Instruction where compilation begins.

STG_MULTIPLE_MICRO_OPS_ADDRESS_RANGE

Address range for MTE stores with multiple uops. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

STORE_MULTIPLE_REGS_ADDRESS_RANGE

Address range for stores with multiple registers. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

SUBOPTIMAL_LDST_RETIRED

suboptimal load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1: half-word, 2: word ...)

OPTIMAL_ALIGN_SELECTOR unsigned int
: Selector to decide the optimal alignment size (1: access width, 2: 4-bytes, 3: 8-bytes, ..., 12: 4-kbytes)

VADDR unsigned int
: The virtual address of the access

SYNC

Called for every synchronization, that is every quantum break. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: The instruction count of this CPU.

LOCAL_QUANTUM unsigned int
: The local quantum of this CPU.

LOCAL_TIME unsigned int
: The local time of this CPU.

SYNC_REASON enum
: Reason for sync.

SYSCALL

System call instruction executed. Fields:

IMM unsigned int
: Immediate value of the system call instruction.

TYPE enum
: System call type.

VADDR unsigned int
: Instruction that caused the system call.

SYSREG_READ

Trace reads from system registers. Fields:

REG_NAME enum
: Register number

VALUE unsigned int
: The value read.

SYSREG_UPDATE32

Triggers when the system updates a register. Fields:

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

SYSREG_UPDATE64

Triggers when the system updates a register. Fields:

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

SYSREG_WRITE

Trace write to system registers. Fields:

REG_ACCESS enum
: Register access status

REG_NAME enum
: Register number

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VALUE unsigned int
: Value written to the register.

UNALIGNED_LDST_RETIRED

Processor unaligned load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1=>half-word, 2->word ...)

NUMBER_OF_BEATS unsigned int
: Number of accesses (beats) in this burst (Total size in bytes is ACCESS_SIZE * NUMBER_OF_BEATS)

VADDR unsigned int
: The virtual address of the access

WAYPOINT

Signals end of basic block execution. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: Origin instructions set

IS_COND bool
: Indicates if this is a conditional waypoint.

PC unsigned int
: Origin address (or 0 if unavailable)

TAKEN bool
: Indicates if this waypoint was taken.

TARGET unsigned int
: Destination address

TARGET_ISET enum
: Destination instructions set

WFE_END

WFE ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFE.

WFE_EVENT_REGISTER

WFE event register status: set/clear, reason. Fields:

INST_COUNT unsigned int
: Ticks count.

REASON enum
: Reason for set/clear. Only REASON==0 (Cleared by WFE) clears the bit, all other reasons set it.

WFE_IGNORED

WFE ignored. Fields:

EVENT bool
: This WFE was ignored because the event register was set.

INST_COUNT unsigned int
: Ticks count when ignoring WFE.

TRAPPED bool
: This WFE was trapped.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

WFE_START

WFE entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFE.

WFI_END

WFI ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFI.

WFI_IGNORED

WFI ignored. Fields:

DISABLED bool
: This WFI was ignored because WFI is disabled.

INST_COUNT unsigned int
: Ticks count when ignoring WFI.

REASON enum
: specifies reason why WFI trace was ignored

TRAPPED bool
: This WFI was trapped.

WFI_START

WFI entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFI.

WFI_WAKEUP

WFI wakeup. Fields:

INST_COUNT unsigned int
: Ticks count when WFI wakeup occurred.

REASON enum
: Reason for wakeup.

XPSR

Changes to the xPSR register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

OLD_VALUE unsigned int
: The old xPSR value

UNKNOWN unsigned int
: Bits within the register that have unknown value.

VALUE unsigned int
: The new xPSR value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.33 ARM_Cortex-M33 - trace

This section describes the trace sources.

ASYNC_MEMORY_FAULT

Context ID Register write. Fields:

FAULT unsigned int
: Fault status

PADDR unsigned int
: Physical Address (or 0 if unavailable)

ATOMIC_END_ACCESS

Bus trace access for atomics. Fields:

ACCESS_FAIL bool
: Memory access failed

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

LOAD_VALUE unsigned int
: Loaded values

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ATOMIC_START_ACCESS

Bus trace access for atomics. Fields:

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ArchMsg.Error.exit_code

Fields:

EXITCODE signed int
: Exit code to be returned

KIND string
: Component kind that invoked the exit code trace

REASON string
: Reason why the exit code trace was invoked (optional)

ArchMsg.Warning.TarmacReconstructor

Unexpected event happened during Tarmac reconstruction. Fields:

REASON string
: Reason

ArchMsg.Warning.Unpredictably Indexed PM Event Register

Fields:

IsDirect unsigned int
: Direct Access rather than indirect

N unsigned int
: Number of Accessible Registers

SEL unsigned int
: Selector

ArchMsg.Warning.branch_to_unaligned_address

Fields:

ADDR unsigned int
: Unaligned branch address

ArchMsg.Warning.dap_csw_bad_size

A write to CM3DAP CSW has an invalid size field. Fields:

DATA unsigned int
: bits[2:0] are size

ArchMsg.Warning.dcimvac_matches_watchpoint

Watchpoints matching an AArch32 DCIMVA are implementation defined.

ArchMsg.Warning.decode_fieldmismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_fracbitsnegative

VCVT Instruction is unpredictable with a negative number of fraction bits.

ArchMsg.Warning.decode_initblock

Instruction is unpredictable in an IT block.

ArchMsg.Warning.decode_initblocknotlast

Instruction is unpredictable in an IT block when not the last.

ArchMsg.Warning.decode_invalid_condition

Instruction is unpredictable when the condition is not AL.

ArchMsg.Warning.decode_invalidfieldcombination

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_invalidvalue

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_mem_hint_unallocated

There is no memory hint allocated to this bit pattern.

ArchMsg.Warning.decode_registermatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registermismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registeroutofrange

Float point instruction is unpredictable indexing beyond the end of the register bank.

ArchMsg.Warning.decode_sbzsbo

Reserved bits in the instruction are not canonical.

ArchMsg.Warning.decode_transactiontoobig

Attempt to perform a memory transaction of more than 128 bytes.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_unpred_other

Instruction is unpredictable.

ArchMsg.Warning.decode_unpreduseofpc

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13inreglist

Instruction is unpredictable with R13 in register-list.

ArchMsg.Warning.decode_useofd16orhigher

Attempt to use register D16 or higher where it is not allowed.

ArchMsg.Warning.decode_writebackandbaseinlist

Instruction is unpredictable with write back when the base register is in the transfer list.

ArchMsg.Warning.decode_zeroregistersinlist

Instruction is unpredictable when no registers are to be transfered.

ArchMsg.Warning.recursive_branch

An instruction is performing a branch that targets the same instruction. If this was intended then a WFI might be more effective. Fields:

PC unsigned int
: Address of instruction causing the branch

ArchMsg.Warning.recursive_exception

An instruction has generated an exception that targets the same instruction. Fields:

CPSR unsigned int
: Processor state

PC unsigned int
: Address of instruction causing the exception

TYPE enum
: Type of exception

ArchMsg.Warning.reentrant_vector_catch

Fields:

ADDR unsigned int
: vector address

ArchMsg.Warning.reserved_it_state

Fields:

ITSTATE unsigned int
: reserved IT state

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.secure_vector_fetch_from_nonsecure

Secure vector table is being fetched from Non-secure memory .

ArchMsg.Warning.sg_in_it_block

Arm recommends that software does not place SG (security gate instruction) inside an IT instruction block.

ArchMsg.Warning.unpredictable_a32_breakpoint

Fields:

ADDR1 unsigned int
: Address

ADDR2 unsigned int
: Breakpoint address

BAS unsigned int
: BAS Field

IS_ADDRESS_MISMATCH_BREAKPOINT bool
: Is Mismatch breakpoint, not a match breakpoint

N unsigned int
: Breakpoint number

OFFSET unsigned int
: Indicates which halfword actually considering into the A32 instruction

ArchMsg.Warning.unpredictable_banked_register_access

Fields:

R unsigned int
: R bit

SYSM unsigned int
: SYSm

ArchMsg.Warning.unpredictable_ccfail_undef

Fields:

OPCODE unsigned int
: opcode of the instruction

VADDR unsigned int
: Virtual address of the instruction being executed

ArchMsg.Warning.unpredictable_ici_bits

ICI bits do not apply or are architecturally unpredictable for instruction.

ArchMsg.Warning.unpredictable_t32_breakpoint

Fields:

ADDR1 unsigned int
: Address

ADDR2 unsigned int
: Breakpoint address

BAS unsigned int
: BAS Field

IS_ADDRESS_MISMATCH_BREAKPOINT bool
: Is Mismatch breakpoint, not a match breakpoint

N unsigned int
: Breakpoint number

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_t32_vector_catch

Fields:

ADDR unsigned int
Address
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_watchpoint_far

Fields:

LOWER_BOUND unsigned int
the lowest address accessed by the instruction that triggered the watchpoint
UPPER_BOUND unsigned int
the highest watchpointed address accessed by that instruction

ArchMsg.Warning.warning_AdvSIMDExpandImmUnexpectedZero

AdvSIMDExpandImm may treat this immediate value as UNPREDICTABLE.

ArchMsg.Warning.warning_ConditionalSMC

SMC instruction has UNPREDICTABLE effects when conditional when combined with traps.

ArchMsg.Warning.warning_access_crosses_page_boundary

Access crosses page boundary.

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault_on_both_pages

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_spanning_different_memory

Fields:

ADDR unsigned int
address of access crossing page
MEMTYPE_PAGE1 signed int
First Page's memory type (0-Normal, 1-Non-Normal)
MEMTYPE_PAGE2 signed int
Second Page's memory type (0-Normal, 1-Non-Normal)

ArchMsg.Warning.warning_access_wraps_around_memory

Fields:

WIDTH unsigned int
Address width

ArchMsg.Warning.warning_bx_from_thumbee

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR unsigned int
Destination address

ArchMsg.Warning.warning_ccsidr_unimplemented_level

Fields:

CSSELR unsigned int
current effective value of CSSELR

ArchMsg.Warning.warning_change_to_ns_when_tge_set

Attempting to change NS to 1 when HCR.TGE = 1.

ArchMsg.Warning.warning_contiguous_bit_check_abort

Fields:

ENTRY_ADDR unsigned int
address of conflicting TLB entry

ArchMsg.Warning.warning_contiguous_bit_error

Fields:

CONTIG_ADDR unsigned int
address of the TLB first read, with which this entry is expected to be contiguous
CONTIG_DATA unsigned int
expected contents based on the entry at CONTIG_ADDR
ENTRY_ADDR unsigned int
address of conflicting TLB entry
ENTRY_DATA unsigned int
contents of conflicting TLB entry

ArchMsg.Warning.warning_cp10_cp11_mismatch

Fields:

EL unsigned int
EL owning the control

ArchMsg.Warning.warning_cp10_cp11_reserved_value

Fields:

CP10 unsigned int
CP10 access
CP11 unsigned int
CP11 access

ArchMsg.Warning.warning_csselr_level_out_of_range

Fields:

IMPLEMENTED unsigned int
number of cache levels implemented
LEVEL unsigned int
written value of CSSELR.Level
TYPE enum
The type of cache selected by CSSELR.InD and TnD

ArchMsg.Warning.warning_debug_flow_control_bits_not_obeyed

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ERROR enum
: type of error resulting

REGISTER enum
: register under consideration

ArchMsg.Warning.warning_debug_register_access_during_reset

Fields:

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

ArchMsg.Warning.warning_decode_cps_inconsistent_fields

The mode setting fields in the CPS instruction are inconsistent.

ArchMsg.Warning.warning_decode_invalid_state

Instruction is unpredictable in current exception-level/security state.

ArchMsg.Warning.warning_default_cacheable_mmu_on

Default cacheable is enabled (HCR.DC = 1) while MMU is enabled.

ArchMsg.Warning.warning_default_cacheable_vmmu_off

Default cacheable is enabled (HCR.DC = 1) while HCR.VM = 0.

ArchMsg.Warning.warning_deprecated_wvr_bit_2_set

Fields:

N unsigned int
: index

WVR unsigned int
: DBGWVR

ArchMsg.Warning.warning_eret_while_software_step_active_pending

Missing ISB between setting MDSCR_EL1.SS and ERET.

ArchMsg.Warning.warning_exclusive_to_non_normal

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_exclusive_to_non_writeback

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_execute_from_device_memory

An attempt was made to execute from Device memory.

ArchMsg.Warning.warning_illegal_cpsr_mode

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NEW_MODE unsigned int
New value of CPSR.M

ArchMsg.Warning.warning_illegal_srs_mode

Fields:

MODE unsigned int
mode for Banked SP

ArchMsg.Warning.warning_implementation_defined_read_debug_register_in_SCS

Read Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

PPB_OFFSET unsigned int
debug register offset

ArchMsg.Warning.warning_implementation_defined_sequential_security_transitions_supported

The behaviour of a sequential instruction fetches that cross from non-secure to secure memory and contain SG instruction is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_implementation_defined_stack_limit_check_supported

It is IMPLEMENTATION DEFINED whether stack pointer limit checking is performed for the this instructions. Fields:

ADDRESS unsigned int
Address of instruction
OPCODE unsigned int
instruction opcode

ArchMsg.Warning.warning_implementation_defined_write_debug_register_in_SCS

Write Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

DATA unsigned int
data attempted to be written
PPB_OFFSET unsigned int
debug register offset

ArchMsg.Warning.warning_invalid_tcr_granule

Fields:

REQUEST signed int
page size requested (or 0 for reserved)
SUBSTITUTE signed int
best guess available page size
TG_ID bool
bits TG0 or TG1

ArchMsg.Warning.warning_load_multiple_user_registers_from_user_mode

An LDM(user registers) instruction executed from user mode.

ArchMsg.Warning.warning_load_pc_from_unaligned

PC loaded from an unaligned location.

ArchMsg.Warning.warning_mte_in_mem_attr_not_supported

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ATTR unsigned int
Memory attribute

ArchMsg.Warning.warning_reserved_breakpoint_state_match

Fields:

IS_BREAKPOINT unsigned int
Is breakpoint, not watchpoint

ArchMsg.Warning.warning_shareability

Unpredictable: combination of the 1st and 2nd stages of translation is Normal Inner Non-Cacheable, Outer Non-Cacheable.

ArchMsg.Warning.warning_software_step_set_while_enabled

MDSCR_EL1.SS set to 1 while software step debug exceptions are enabled.

ArchMsg.Warning.warning_thumb_instruction_wraps_around_memory

Thumb instruction warps around memory.

ArchMsg.Warning.warning_ttbr_sbz_bits_are_not_zero

Fields:

MSB unsigned int
x-1
TTBR enum
which TTBR register

ArchMsg.Warning.warning_unaligned_address_dbgdtrrx_write

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_address_dbgdtrtx_read

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_device

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_strongly_ordered

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unknown_sau_rnr

SAU_RNR was set to an unsupported value.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_AIRCR_PRIP_and_BFHFNMINP

The effect of setting both AIRCR.BFHFNMINP and AIRCR.PRIP to 1 is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTCLRACTIVE_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTCLRACTIVE if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_and_SYSRESETREQ

When the processor is halted in Debug state, if a write to the register writes a 1 to both VECTRESET and SYSRESETREQ, the behavior is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTRESET if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_incorrect_VKEY

The value 0x05FA must be written to AIRCR.VKEY, otherwise the register write is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_EXE_RETURN_Reserved_Bit

EXC_RETURN[23:7] are reserved with the special condition that all bits should be written as one. Values other than all 1s are UNPREDICTABLE. Fields:

EXC_RETURN `unsigned int`
: Special PC value

SBO_MASK `unsigned int`
: Which bits Should Be One to not UNPRED

ArchMsg.Warning.warning_unpredictable_change_to_DEMCR_MON_STEP_at_insufficient_priority

The effect of changing DEMCR.MON_STEP at an execution priority that is lower than the priority of the DebugMonitor exception is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_change_to_priority_of_active_exception

Changing the priority of an active exception is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_clear_lspact

Fields:

DRVEXC_TAKEN `unsigned int`
: derived exception taken not pended

REACHED_LAST_STORE `unsigned int`
: only the last faulted

ArchMsg.Warning.warning_unpredictable_dbg_exit_unaligned_dlr

Debug exit to AArch32 with DLR[0] set to 1. PC[0] can be set to 0 or to DLR[0].

ArchMsg.Warning.warning_unpredictable_exception_catch

Generation of an exception catch event is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_inconsistent_state

state on exception return is unpredictable.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_exception_return_instruction

exception return instruction is unpredictable.

ArchMsg.Warning.warning_unpredictable_in_debug_state

Instruction is UNPREDICTABLE when executed in debug state.

ArchMsg.Warning.warning_unpredictable_mair_encoding

Fields:

ATTR unsigned int
: Memory attribute

ArchMsg.Warning.warning_unpredictable_pmu_counter_access

Access to PMU counters from non-secure EL0 or EL1 is UNPREDICTABLE with MDCR_EL2.HPMN set to 0.

ArchMsg.Warning.warning_unpredictable_prioritization_breakpoint_match_vector_catch

Two events with the same priority occurred at the same instruction: Address Matching Vector Catch debug event, and Breakpoint debug event It is constrained unpredictable which is taken.

ArchMsg.Warning.warning_unpredictable_stack_selection

UNPREDICTABLE state on context switch.

ArchMsg.Warning.warning_unpredictable_tsize_out_of_range

Fields:

TCR enum
: Which TCR

TSIZE signed int
: Value of TSize

TSIZE_MAX signed int
: The maximum allowed value

TTBR unsigned int
: Which TTBR region

ArchMsg.Warning.warning_unpredictable_unaligned_pc_as_base_register

The PC value must be word-aligned, otherwise the behavior of the instruction is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_unaligned_pop_stack

Fields:

FRAME_ADDR unsigned int
: unaligned stack frame address

ArchMsg.Warning.warning_unpredictable_vmsa_memattrib

Fields:

TEXCB unsigned int
: texcb value

TEX_REMAP bool
: use tex remap

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_vtcr_t0sz_sl0

Fields:

SL0 signed int
: VTCR.SL0, starting level for stage 2 translation table walks

T0SZ signed int
: VTCR.T0SZ, required input address range

ArchMsg.Warning.warning_unpredictable_write_DHCSR

Fields:

NEW unsigned int
: data attempted to be written

OLD unsigned int
: previous value

ArchMsg.Warning.warning_unpredictable_write_SHCSR

UNPREDICTABLE setting a pending bit to 1 for an exception with priority >= execution priority.

ArchMsg.Warning.warning_unsupported_access_to_memory_mapped_register

Access to memory mapped register is unsupported.

ArchMsg.Warning.warning_user_jmcr_access

Fields:

WRITE unsigned int
: access is write

ArchMsg.Warning.warning_wcr_mask_and_bas

DBGWCR_EL1.MASK is non-zero and BAS is not 0xff.

ArchMsg.Warning.warning_wcr_mask_reserved

DBGWCR_ELn.MASK is set to a reserved value.

ArchMsg.Warning.warning_wcr_non_configuous_bas

DBGWCR_EL1.BAS has non-contiguous set of ones.

ArchMsg.Warning.warning_wvr_masked_bit_not_zero

DBGWVRn_EL1.VA contains a bit masked by DBGWCRn_EL1.MASK that is not zero.

ArchMsg.warning_unpred_system_register_access

Fields:

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

BRANCH_MISPREDICT

Simulating branch mispredict. Fields:

PC unsigned int
: Origin address (or 0 if unavailable)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

BRA_DIR

Direct branches, to immediate address. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR

Indirect branches, perhaps to a register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR_CCFAIL

Indirect branches fail, perhaps to a register.

BRA_RET_CCFAIL

Branche return fail, perhaps to a register.

CCFAIL

Conditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CCFAIL_UNC

Unconditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS

Conditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS_UNC

Unconditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CODE_CACHE_MAINT

Code cache maintenance. Fields:

END_INCL unsigned int
: Inclusive end addres for by VA or PA requests

NS bool
: Non-secure world for PA

START unsigned int
: Start address for by VA or PA requests

TYPE enum
: Request type.

COMPILE_BLOCK_END

Last instruction of basic block translated. Fields:

VADDR unsigned int
: Address of next instruction after this basic block

COMPILE_BLOCK_START

First instruction of basic block translated. Fields:

VADDR unsigned int
: Address of first instruction in basic block

COMPILE_INST

ARM instruction compiled. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DISASS string
: Disassembly of the instruction.

ISET enum
: The instruction set of this instruction.

ITSTATE unsigned int
: The ITSTATE current for the instruction.

OPCODE unsigned int
: The opcode of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

CONTEXTIDR

Context ID Register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

NS bool
: Secure or nonsecure banked register is accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

CONTEXT_SYNC

Called for every context synchronization event. Fields:

ARCH_CSE bool
: Architectural Context Synchronization Event

SYSREG_SYNC bool
: System register synchronization event

CORE_ENDIAN

Core BE8 Big-Endian state changed. Fields:

BE8 bool
: Core BE8 Big-Endian state

CORE_INFO

Static processor attributes. Only triggered by a call to DumpState(). Fields:

ARCH_PROFILE enum
: The architecture profile of the core.

CLUSTER_ID unsigned int
: The cluster ID of this processor.

CORE_NUM unsigned int
: The number of this core in an MP processor.

DETAILED_CORE_FEATURES enum
: Detailed core features as a bitfield. Enum values are bitmasks to extract the info from the field

FPU_VERSION enum
: The VFP version implemented by the core.

MEM_ARCH enum
: The memory architecture of the core.

MIDR_EL1 unsigned int
: MIDR_EL1 value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NUM_CORES unsigned int
: The number of cores in this MP processor.

QUANTUM_SIZE unsigned int
: The default quantum size of the core.

SECURITY_FEATURES bool
: Does the core have security features?

CORE_LOADS

Processor load accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CORE_REGS

Changes of the core registers R0 to R14. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ID unsigned int
: The register number, 0 to 14.

OLD_VALUE unsigned int
: The old value overwritten.

PHYS_ID enum
: The physical register accessed.

VALUE unsigned int
: The new value written to the register.

CORE_STORES

Processor store accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CRYPTO_SPEC

Every crypto instruction speculatively executed.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DATA_CACHE_SET_ALLOC_TAGS

Set allocation tags from DC GVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of memory in bytes.

TAG unsigned int
: The value of the tag being set.

VADDR unsigned int
: Virtual address used for setting allocation tags.

DATA_CACHE_ZERO

Zero and invalidate cache line from DC ZVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of zero'd memory in bytes.

VADDR unsigned int
: Virtual address of the zero'd cache line.

DEBUG_EVENT

Hardware debug support event. Fields:

EVENT enum
: Description of event

VALUE unsigned int
: data value

DWT_MATCH

DWT comparator matches. Fields:

NUM unsigned int
: DWT comparator number

TYPE enum
: DWT comparator configuration

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

END_COMPILE

Compilation end. Fields:

END_OF_PAGE_COUNT unsigned int
: Number of basic blocks exited due to page boundary since START_COMPILE.

FETCHFAIL_COUNT unsigned int
: Number of basic blocks exited due to ifetch failure START_COMPILE.

INST_COUNT unsigned int
: Number of instructions compiled since START_COMPILE.

NONSEQ_COUNT unsigned int
: Number of basic blocks exited due to non-sequential instructions since START_COMPILE.

PAGE_STRADDLE_COUNT unsigned int
: Number of basic blocks exited due to unaligned instructions crossing page since START_COMPILE.

EXCEPTION

Exceptions which are taken. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: The instruction set of the processor when the exception occurred.

LR unsigned int
: The value assigned to the link register.

NS enum
: The core's non-secure bit.

PC unsigned int
: The location where the exception occurred.

TARGET_ISET enum
: The instruction set of the exception handler code.

TARGET_PC unsigned int
: The address the exception branches to.

VECTOR enum
: The exception vector.

EXCEPTION_END

Every exception completed. Fields:

NS enum
: The core's non-secure bit.

EXCEPTION_ENTRY

Event marking the entry of an exception. All pushes of register to the stack will follow this event. Fields:

PC unsigned int
: The location where the exception occurred.

VECTOR enum
: The exception that occurred.

EXCEPTION_RAISE

Every exception raised.

EXCEPTION_RETURN

Marks the end of an exception, but on an M core does not indicate a program flow change (branch). In most situations it will be followed by an indirect branch. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

PC unsigned int
: The address of the branch instruction.

EXCEPTION_RETURN_PREBRANCH

Exception return event that must occur before PC and processor state have updated.

EXCEPTION_START

Every exception started. Fields:

NS enum
: The core's non-secure bit.

EXCEPTION_VECTOR_FETCH

I-side vector fetch for M-class exception. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Result of fetch if OK

RESPONSE enum
: 0=Aborted, 1=OK

VADDR unsigned int
: Where the vector is fetched from (including VTOR)

FP_STATE

Floating point state. Fields:

FPSR unsigned int
: FPSR register value.

MASK unsigned int
: The mask indicating updated cumulative bits.

HLT

HLT instruction occurred. Fields:

IMM16 unsigned int
: The 16-bit immediate value encoded in HLT instruction.

INFO_EXCEPTION_REASON

Provide information of real cause of exception. Fields:

FaultCause enum
: Which bit of which V7M/V8M-mainline FSR/HFSR would have been set.

PC unsigned int
: The location where the exception occurred.

PHASE enum
: What we are doing with the exception.

REASONS enum
: Why we are doing it.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SecurityState enum
: Privacy State of exception

VECTOR enum
: The exception that occurred.

INFO_FP_CONTEXT

Inform when ExecuteFPCheck writes CONTROL.FPCA and FPSCR etc. Fields:

WHAT enum
: What we are doing.

INFO_STACKING

Provide information of real cause of exception. Fields:

PHASE enum
: Are we starting or ending a sequence of stack writes.

REASONS enum
: Why we are doing it.

INST

Every instruction executed. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DEBUG_STATE bool
: The instruction is executed in debug state.

DISASS string
: Disassembly of instruction.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The current instruction set.

ITSTATE unsigned int
: The current ITSTATE.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

MODE enum
: The mode the core is in.

NS unsigned int
: The core's non-secure bit.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

OPCODE unsigned int
: The opcode of the instruction.

PADDR unsigned int
: The physical address of the instruction.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_END

Every instruction completed.

INST_START

Every instruction started. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

INST_STRADDLE

Instruction straddles boundary. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

ISB

ISB instruction occurred.

ITM

Instrumentation Trace Macrocell. Fields:

ITM_PACKET_TYPE enum
: ITM and DWT packets type.

PACKET_HEADER unsigned int
: ITM Packet Header.

PACKET_PAYLOAD unsigned int
: ITM Packet Payload.

LOAD_MULTIPLE_REGS_LIST

List of destination registers (not source) of a multiple registers load instruction. Fields:

REG string
: Register to add into the list

LOCAL_MONITOR

Local monitor activity. Fields:

PADDR unsigned int
: Local Monitor Address

State enum
: State of the monitor (Open/Exclusive)

LOCKUP_CYCLE

This event is triggered if the core enters or stays in the lock-up state. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CAUSE enum
: What causes the lockup?.

LOCKUP_ENTRY

This event is triggered if the core enters the lock-up state. Fields:

PC unsigned int
: The PC when the lockup state is entered.

LOCKUP_EXIT

This event is triggered if the core leaves the lock-up state.

MODE_CHANGE

Mode change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: The new mode.

OLD_MODE enum
: The old mode.

MPU_TRANS

Address translation information. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

EXEC_PERM enum
: Execution Permission

IDAU_REG_NUM enum
: IDAU region-number or code

INNERCACHE_RA bool
: Is the inner cache allocate on read

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

MEMTYPE enum
: Memory type.

NSDESC enum
: Security-attributes of the transaction.

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

PADDR unsigned int
: Address of the access.

PAGESIZE unsigned int
: Page size as log2(size).

READ_PERM enum
: Read Permission

REG_NUM signed int
: MPU region-number 0-15 (else 0xFF=none, 0xBB/0xDD=background/default)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SAU_REG_NUM enum
: SAU region-number or code

SAU_RGN_TYPE enum
: Privacy Region Type

SH enum
: Shareability

SIDE enum
: Inst / Data.

WRITE_PERM enum
: Write Permission

PERIODIC

Called for every quantum. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

PC unsigned int
: The address of the next instruction to be executed on this CPU.

PPB_READ_ACCESS

Trace reads from the core to memory mapped registers in the PPB (private peripheral bus) address range. Fields:

ACCESS_SIZE unsigned int
: Logarithm of the access size: 0=byte, 1=halfword, 2=word, 3=doubleword, ...

BURST_LEN unsigned int
: The number of data transfers (beats) in this burst.

DATA unsigned int
: The data transferred.

INNER_ATTRIBUTES unsigned int
: The inner attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

INST bool
: Is an instruction fetch.

LOCK enum
: Normal / Locked / Exclusive.

MASTER_ID unsigned int
: The AXI master ID.

NS bool
: Is a nonsecure access.

OUTER_ATTRIBUTES unsigned int
: The outer attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

PADDR unsigned int
: Physical address of access.

PRIV bool
: Is privileged access.

RESPONSE enum
: Whether the transaction was successful, or an error occurred.

USER_FLAGS unsigned int
: Core specific additional signals.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PPB_WRITE_ACCESS

Trace writes from the core to memory mapped registers in the PPB (private peripheral bus) address range. Fields:

ACCESS_SIZE unsigned int
: Logarithm of the access size: 0=byte, 1=halfword, 2=word, 3=doubleword, ...

BURST_LEN unsigned int
: The number of data transfers (beats) in this burst.

DATA unsigned int
: The data transferred.

INNER_ATTRIBUTES unsigned int
: The inner attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

INST bool
: Is an instruction fetch.

LOCK enum
: Normal / Locked / Exclusive.

MASTER_ID unsigned int
: The AXI master ID.

NS bool
: Is a nonsecure access.

OUTER_ATTRIBUTES unsigned int
: The outer attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

PADDR unsigned int
: Physical address of access.

PRIV bool
: Is privileged access.

RESPONSE enum
: Whether the transaction was successful, or an error occurred.

USER_FLAGS unsigned int
: Core specific additional signals.

PRELOAD_DATA

Data preload from PLD instruction. Fields:

VADDR unsigned int
: Virtual address of the data that should be preloaded.

PRELOAD_INST

Instruction preload from PLI instruction. Fields:

VADDR unsigned int
: Virtual address of the instruction that should be preloaded.

PRE_CORE_LOAD

Trace just before a core load. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_CORE_STORE

Trace just before a core store. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PSTATE_PACTIVE_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

PSTATE_PACTIVE_OPMODE_TRANSITION_DENIED

Trace the denied pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_POWER_TRANSITION

Trace pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

PSTATE_PACTIVE_POWER_TRANSITION_DENIED

Trace the denied pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_THREAD_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions for threads on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OLD_STATE enum
: Operating mode before the transition

RUN_STATE

Run state transition. Fields:

INST_COUNT unsigned int
: Ticks count at point of transition.

NEW enum
: New run state.

OLD enum
: Old run state.

SEMIHOSTING_CALL

Call of a semihost function occurred. Fields:

PC unsigned int
: The program counter after the semihosting call.

SEMIHOSTING_PRECALL

About to call semihost. Fields:

EL2 bool
: Use EL2 translation regime.

NS bool
: Is Non-Secure.

REG_WIDTH unsigned int
: The current register width in bytes.

SIGNAL

External signal state change. Fields:

SIGNAL enum
: Signal that changed

STATE bool
: Signal asserted state

START_COMPILE

Compilation started. Fields:

VADDR unsigned int
: Instruction where compilation begins.

STG_MULTIPLE_MICRO_OPS_ADDRESS_RANGE

Address range for MTE stores with multiple uops. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

STORE_MULTIPLE_REGS_ADDRESS_RANGE

Address range for stores with multiple registers. Fields:

ADDR_END unsigned int
: Final address.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR_START unsigned int
: Starting address.

SUBOPTIMAL_LDST_RETIRED

suboptimal load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1: half-word, 2: word ...)

OPTIMAL_ALIGN_SELECTOR unsigned int
: Selector to decide the optimal alignment size (1: access width, 2: 4-bytes, 3: 8-bytes, ..., 12: 4-kbytes)

VADDR unsigned int
: The virtual address of the access

SYNC

Called for every synchronization, that is every quantum break. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

LOCAL_QUANTUM unsigned int
: The local quantum of this CPU.

LOCAL_TIME unsigned int
: The local time of this CPU.

SYNC_REASON enum
: Reason for sync.

SYSCALL

System call instruction executed. Fields:

IMM unsigned int
: Immediate value of the system call instruction.

TYPE enum
: System call type.

VADDR unsigned int
: Instruction that caused the system call.

SYSREG_READ

Trace reads from system registers. Fields:

REG_NAME enum
: Register number

VALUE unsigned int
: The value read.

SYSREG_UPDATE32

Triggers when the system updates a register. Fields:

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SYSREG_UPDATE64

Triggers when the system updates a register. Fields:

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

SYSREG_WRITE

Trace write to system registers. Fields:

REG_ACCESS enum
: Register access status

REG_NAME enum
: Register number

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: Value written to the register.

UNALIGNED_LDST_RETIRED

Processor unaligned load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1=>half-word, 2->word ...)

NUMBER_OF_BEATS unsigned int
: Number of accesses (beats) in this burst (Total size in bytes is ACCESS_SIZE * NUMBER_OF_BEATS)

VADDR unsigned int
: The virtual address of the access

VFP_D_REGS

VFP/NEON D 64 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_Q_REGS

VFP/NEON Q 128 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding byte in the register. For example, 0x1 would indicate a write to VALUE[7:0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_SYS_REGS

Writes to the VFP/NEON units system registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: Which VFP system register is written.

MASK unsigned int
: The mask indicating updated cumulative bits.

OLD_VALUE unsigned int
: The register's old value overwritten.

VALUE unsigned int
: The new value written to the VFP system register.

VFP_S_REGS

VFP/NEON S 32 bit register write. Fields:

ALIASING enum
: Type of register aliasing used.

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

WAYPOINT

Signals end of basic block execution. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: Origin instructions set

IS_COND bool
: Indicates if this is a conditional waypoint.

PC unsigned int
: Origin address (or 0 if unavailable)

TAKEN bool
: Indicates if this waypoint was taken.

TARGET unsigned int
: Destination address

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

TARGET_ISET enum
Destination instructions set

WFE_END

WFE ended. Fields:

INST_COUNT unsigned int
Ticks count when leaving WFE.

WFE_EVENT_REGISTER

WFE event register status: set/clear, reason. Fields:

INST_COUNT unsigned int
Ticks count.
REASON enum
Reason for set/clear. Only REASON==0 (Cleared by WFE) clears the bit, all other reasons set it.

WFE_IGNORED

WFE ignored. Fields:

EVENT bool
This WFE was ignored because the event register was set.
INST_COUNT unsigned int
Ticks count when ignoring WFE.
TRAPPED bool
This WFE was trapped.

WFE_START

WFE entered. Fields:

INST_COUNT unsigned int
Ticks count when entering WFE.

WFI_END

WFI ended. Fields:

INST_COUNT unsigned int
Ticks count when leaving WFI.

WFI_IGNORED

WFI ignored. Fields:

DISABLED bool
This WFI was ignored because WFI is disabled.
INST_COUNT unsigned int
Ticks count when ignoring WFI.
REASON enum
specifies reason why WFI trace was ignored
TRAPPED bool
This WFI was trapped.

WFI_START

WFI entered. Fields:

INST_COUNT unsigned int
Ticks count when entering WFI.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

WFI_WAKEUP

WFI wakeup. Fields:

INST_COUNT unsigned int
: Ticks count when WFI wakeup occurred.

REASON enum
: Reason for wakeup.

XPSR

Changes to the xPSR register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

OLD_VALUE unsigned int
: The old xPSR value

UNKNOWN unsigned int
: Bits within the register that have unknown value.

VALUE unsigned int
: The new xPSR value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.34 ARM_Cortex-M35P - trace

This section describes the trace sources.

ASYNC_MEMORY_FAULT

Context ID Register write. Fields:

FAULT unsigned int
: Fault status

PADDR unsigned int
: Physical Address (or 0 if unavailable)

ATOMIC_END_ACCESS

Bus trace access for atomics. Fields:

ACCESS_FAIL bool
: Memory access failed

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

LOAD_VALUE unsigned int
: Loaded values

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ATOMIC_START_ACCESS

Bus trace access for atomics. Fields:

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ArchMsg.Error.exit_code

Fields:

EXITCODE signed int
: Exit code to be returned

KIND string
: Component kind that invoked the exit code trace

REASON string
: Reason why the exit code trace was invoked (optional)

ArchMsg.Warning.TarmacReconstructor

Unexpected event happened during Tarmac reconstruction. Fields:

REASON string
: Reason

ArchMsg.Warning.Unpredictably Indexed PM Event Register

Fields:

IsDirect unsigned int
: Direct Access rather than indirect

N unsigned int
: Number of Accessible Registers

SEL unsigned int
: Selector

ArchMsg.Warning.branch_to_unaligned_address

Fields:

ADDR unsigned int
: Unaligned branch address

ArchMsg.Warning.dap_csw_bad_size

A write to CM3DAP CSW has an invalid size field. Fields:

DATA unsigned int
: bits[2:0] are size

ArchMsg.Warning.dcimvac_matches_watchpoint

Watchpoints matching an AArch32 DCIMVA are implementation defined.

ArchMsg.Warning.decode_fieldmismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_fracbitsnegative

VCVT Instruction is unpredictable with a negative number of fraction bits.

ArchMsg.Warning.decode_initblock

Instruction is unpredictable in an IT block.

ArchMsg.Warning.decode_initblocknotlast

Instruction is unpredictable in an IT block when not the last.

ArchMsg.Warning.decode_invalid_condition

Instruction is unpredictable when the condition is not AL.

ArchMsg.Warning.decode_invalidfieldcombination

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_invalidvalue

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_mem_hint_unallocated

There is no memory hint allocated to this bit pattern.

ArchMsg.Warning.decode_registermatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registermismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registeroutofrange

Float point instruction is unpredictable indexing beyond the end of the register bank.

ArchMsg.Warning.decode_sbzsbo

Reserved bits in the instruction are not canonical.

ArchMsg.Warning.decode_transactiontoobig

Attempt to perform a memory transaction of more than 128 bytes.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_unpred_other

Instruction is unpredictable.

ArchMsg.Warning.decode_unpreduseofpc

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13inreglist

Instruction is unpredictable with R13 in register-list.

ArchMsg.Warning.decode_useofd16orhigher

Attempt to use register D16 or higher where it is not allowed.

ArchMsg.Warning.decode_writebackandbaseinlist

Instruction is unpredictable with write back when the base register is in the transfer list.

ArchMsg.Warning.decode_zeroregistersinlist

Instruction is unpredictable when no registers are to be transfered.

ArchMsg.Warning.recursive_branch

An instruction is performing a branch that targets the same instruction. If this was intended then a WFI might be more effective. Fields:

PC unsigned int
: Address of instruction causing the branch

ArchMsg.Warning.recursive_exception

An instruction has generated an exception that targets the same instruction. Fields:

CPSR unsigned int
: Processor state

PC unsigned int
: Address of instruction causing the exception

TYPE enum
: Type of exception

ArchMsg.Warning.reentrant_vector_catch

Fields:

ADDR unsigned int
: vector address

ArchMsg.Warning.reserved_it_state

Fields:

ITSTATE unsigned int
: reserved IT state

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.secure_vector_fetch_from_nonsecure

Secure vector table is being fetched from Non-secure memory .

ArchMsg.Warning.sg_in_it_block

Arm recommends that software does not place SG (security gate instruction) inside an IT instruction block.

ArchMsg.Warning.unpredictable_a32_breakpoint

Fields:

ADDR1 unsigned int
: Address

ADDR2 unsigned int
: Breakpoint address

BAS unsigned int
: BAS Field

IS_ADDRESS_MISMATCH_BREAKPOINT bool
: Is Mismatch breakpoint, not a match breakpoint

N unsigned int
: Breakpoint number

OFFSET unsigned int
: Indicates which halfword actually considering into the A32 instruction

ArchMsg.Warning.unpredictable_banked_register_access

Fields:

R unsigned int
: R bit

SYSM unsigned int
: SYSm

ArchMsg.Warning.unpredictable_ccfail_undef

Fields:

OPCODE unsigned int
: opcode of the instruction

VADDR unsigned int
: Virtual address of the instruction being executed

ArchMsg.Warning.unpredictable_ici_bits

ICI bits do not apply or are architecturally unpredictable for instruction.

ArchMsg.Warning.unpredictable_t32_breakpoint

Fields:

ADDR1 unsigned int
: Address

ADDR2 unsigned int
: Breakpoint address

BAS unsigned int
: BAS Field

IS_ADDRESS_MISMATCH_BREAKPOINT bool
: Is Mismatch breakpoint, not a match breakpoint

N unsigned int
: Breakpoint number

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_t32_vector_catch

Fields:

ADDR unsigned int
Address
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_watchpoint_far

Fields:

LOWER_BOUND unsigned int
the lowest address accessed by the instruction that triggered the watchpoint
UPPER_BOUND unsigned int
the highest watchpointed address accessed by that instruction

ArchMsg.Warning.warning_AdvSIMDExpandImmUnexpectedZero

AdvSIMDExpandImm may treat this immediate value as UNPREDICTABLE.

ArchMsg.Warning.warning_ConditionalSMC

SMC instruction has UNPREDICTABLE effects when conditional when combined with traps.

ArchMsg.Warning.warning_access_crosses_page_boundary

Access crosses page boundary.

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault_on_both_pages

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_spanning_different_memory

Fields:

ADDR unsigned int
address of access crossing page
MEMTYPE_PAGE1 signed int
First Page's memory type (0-Normal, 1-Non-Normal)
MEMTYPE_PAGE2 signed int
Second Page's memory type (0-Normal, 1-Non-Normal)

ArchMsg.Warning.warning_access_wraps_around_memory

Fields:

WIDTH unsigned int
Address width

ArchMsg.Warning.warning_bx_from_thumbee

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR unsigned int
Destination address

ArchMsg.Warning.warning_ccsidr_unimplemented_level

Fields:

CSSELR unsigned int
current effective value of CSSELR

ArchMsg.Warning.warning_change_to_ns_when_tge_set

Attempting to change NS to 1 when HCR.TGE = 1.

ArchMsg.Warning.warning_contiguous_bit_check_abort

Fields:

ENTRY_ADDR unsigned int
address of conflicting TLB entry

ArchMsg.Warning.warning_contiguous_bit_error

Fields:

CONTIG_ADDR unsigned int
address of the TLB first read, with which this entry is expected to be contiguous
CONTIG_DATA unsigned int
expected contents based on the entry at CONTIG_ADDR
ENTRY_ADDR unsigned int
address of conflicting TLB entry
ENTRY_DATA unsigned int
contents of conflicting TLB entry

ArchMsg.Warning.warning_cp10_cp11_mismatch

Fields:

EL unsigned int
EL owning the control

ArchMsg.Warning.warning_cp10_cp11_reserved_value

Fields:

CP10 unsigned int
CP10 access
CP11 unsigned int
CP11 access

ArchMsg.Warning.warning_csselr_level_out_of_range

Fields:

IMPLEMENTED unsigned int
number of cache levels implemented
LEVEL unsigned int
written value of CSSELR.Level
TYPE enum
The type of cache selected by CSSELR.InD and TnD

ArchMsg.Warning.warning_debug_flow_control_bits_not_obeyed

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ERROR enum
: type of error resulting

REGISTER enum
: register under consideration

ArchMsg.Warning.warning_debug_register_access_during_reset

Fields:

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

ArchMsg.Warning.warning_decode_cps_inconsistent_fields

The mode setting fields in the CPS instruction are inconsistent.

ArchMsg.Warning.warning_decode_invalid_state

Instruction is unpredictable in current exception-level/security state.

ArchMsg.Warning.warning_default_cacheable_mmu_on

Default cacheable is enabled (HCR.DC = 1) while MMU is enabled.

ArchMsg.Warning.warning_default_cacheable_vmmu_off

Default cacheable is enabled (HCR.DC = 1) while HCR.VM = 0.

ArchMsg.Warning.warning_deprecated_wvr_bit_2_set

Fields:

N unsigned int
: index

WVR unsigned int
: DBGWVR

ArchMsg.Warning.warning_eret_while_software_step_active_pending

Missing ISB between setting MDSCR_EL1.SS and ERET.

ArchMsg.Warning.warning_exclusive_to_non_normal

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_exclusive_to_non_writeback

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_execute_from_device_memory

An attempt was made to execute from Device memory.

ArchMsg.Warning.warning_illegal_cpsr_mode

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NEW_MODE unsigned int
New value of CPSR.M

ArchMsg.Warning.warning_illegal_srs_mode

Fields:

MODE unsigned int
mode for Banked SP

ArchMsg.Warning.warning_implementation_defined_read_debug_register_in_SCS

Read Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

PPB_OFFSET unsigned int
debug register offset

ArchMsg.Warning.warning_implementation_defined_sequential_security_transitions_supported

The behaviour of a sequential instruction fetches that cross from non-secure to secure memory and contain SG instruction is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_implementation_defined_stack_limit_check_supported

It is IMPLEMENTATION DEFINED whether stack pointer limit checking is performed for the this instructions. Fields:

ADDRESS unsigned int
Address of instruction
OPCODE unsigned int
instruction opcode

ArchMsg.Warning.warning_implementation_defined_write_debug_register_in_SCS

Write Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

DATA unsigned int
data attempted to be written
PPB_OFFSET unsigned int
debug register offset

ArchMsg.Warning.warning_invalid_tcr_granule

Fields:

REQUEST signed int
page size requested (or 0 for reserved)
SUBSTITUTE signed int
best guess available page size
TG_ID bool
bits TG0 or TG1

ArchMsg.Warning.warning_load_multiple_user_registers_from_user_mode

An LDM(user registers) instruction executed from user mode.

ArchMsg.Warning.warning_load_pc_from_unaligned

PC loaded from an unaligned location.

ArchMsg.Warning.warning_mte_in_mem_attr_not_supported

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ATTR unsigned int
Memory attribute

ArchMsg.Warning.warning_reserved_breakpoint_state_match

Fields:

IS_BREAKPOINT unsigned int
Is breakpoint, not watchpoint

ArchMsg.Warning.warning_shareability

Unpredictable: combination of the 1st and 2nd stages of translation is Normal Inner Non-Cacheable, Outer Non-Cacheable.

ArchMsg.Warning.warning_software_step_set_while_enabled

MDSCR_EL1.SS set to 1 while software step debug exceptions are enabled.

ArchMsg.Warning.warning_thumb_instruction_wraps_around_memory

Thumb instruction warps around memory.

ArchMsg.Warning.warning_ttbr_sbz_bits_are_not_zero

Fields:

MSB unsigned int
x-1
TTBR enum
which TTBR register

ArchMsg.Warning.warning_unaligned_address_dbgdtrrx_write

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_address_dbgdtrtx_read

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_device

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_strongly_ordered

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unknown_sau_rnr

SAU_RNR was set to an unsupported value.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_AIRCR_PRIP_and_BFHFNMINP

The effect of setting both AIRCR.BFHFNMINP and AIRCR.PRIP to 1 is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTCLRACTIVE_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTCLRACTIVE if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_and_SYSRESETREQ

When the processor is halted in Debug state, if a write to the register writes a 1 to both VECTRESET and SYSRESETREQ, the behavior is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTRESET if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_incorrect_VKEY

The value 0x05FA must be written to AIRCR.VKEY, otherwise the register write is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_EXE_RETURN_Reserved_Bit

EXC_RETURN[23:7] are reserved with the special condition that all bits should be written as one. Values other than all 1s are UNPREDICTABLE. Fields:

EXC_RETURN `unsigned int`
: Special PC value

SBO_MASK `unsigned int`
: Which bits Should Be One to not UNPRED

ArchMsg.Warning.warning_unpredictable_change_to_DEMCR_MON_STEP_at_insufficient_priority

The effect of changing DEMCR.MON_STEP at an execution priority that is lower than the priority of the DebugMonitor exception is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_change_to_priority_of_active_exception

Changing the priority of an active exception is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_clear_lspact

Fields:

DRVEXC_TAKEN `unsigned int`
: derived exception taken not pended

REACHED_LAST_STORE `unsigned int`
: only the last faulted

ArchMsg.Warning.warning_unpredictable_dbg_exit_unaligned_dlr

Debug exit to AArch32 with DLR[0] set to 1. PC[0] can be set to 0 or to DLR[0].

ArchMsg.Warning.warning_unpredictable_exception_catch

Generation of an exception catch event is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_inconsistent_state

state on exception return is unpredictable.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_exception_return_instruction

exception return instruction is unpredictable.

ArchMsg.Warning.warning_unpredictable_in_debug_state

Instruction is UNPREDICTABLE when executed in debug state.

ArchMsg.Warning.warning_unpredictable_mair_encoding

Fields:

ATTR unsigned int
: Memory attribute

ArchMsg.Warning.warning_unpredictable_pmu_counter_access

Access to PMU counters from non-secure EL0 or EL1 is UNPREDICTABLE with MDCR_EL2.HPMN set to 0.

ArchMsg.Warning.warning_unpredictable_prioritization_breakpoint_match_vector_catch

Two events with the same priority occurred at the same instruction: Address Matching Vector Catch debug event, and Breakpoint debug event It is constrained unpredictable which is taken.

ArchMsg.Warning.warning_unpredictable_stack_selection

UNPREDICTABLE state on context switch.

ArchMsg.Warning.warning_unpredictable_tsize_out_of_range

Fields:

TCR enum
: Which TCR

TSIZE signed int
: Value of TSize

TSIZE_MAX signed int
: The maximum allowed value

TTBR unsigned int
: Which TTBR region

ArchMsg.Warning.warning_unpredictable_unaligned_pc_as_base_register

The PC value must be word-aligned, otherwise the behavior of the instruction is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_unaligned_pop_stack

Fields:

FRAME_ADDR unsigned int
: unaligned stack frame address

ArchMsg.Warning.warning_unpredictable_vmsa_memattrib

Fields:

TEXCB unsigned int
: texcb value

TEX_REMAP bool
: use tex remap

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_vtcr_t0sz_sl0

Fields:

SL0 signed int
: VTCR.SL0, starting level for stage 2 translation table walks

T0SZ signed int
: VTCR.T0SZ, required input address range

ArchMsg.Warning.warning_unpredictable_write_DHCSR

Fields:

NEW unsigned int
: data attempted to be written

OLD unsigned int
: previous value

ArchMsg.Warning.warning_unpredictable_write_SHCSR

UNPREDICTABLE setting a pending bit to 1 for an exception with priority >= execution priority.

ArchMsg.Warning.warning_unsupported_access_to_memory_mapped_register

Access to memory mapped register is unsupported.

ArchMsg.Warning.warning_user_jmcr_access

Fields:

WRITE unsigned int
: access is write

ArchMsg.Warning.warning_wcr_mask_and_bas

DBGWCR_EL1.MASK is non-zero and BAS is not 0xff.

ArchMsg.Warning.warning_wcr_mask_reserved

DBGWCR_ELn.MASK is set to a reserved value.

ArchMsg.Warning.warning_wcr_non_configuous_bas

DBGWCR_EL1.BAS has non-contiguous set of ones.

ArchMsg.Warning.warning_wvr_masked_bit_not_zero

DBGWVRn_EL1.VA contains a bit masked by DBGWCRn_EL1.MASK that is not zero.

ArchMsg.warning_unpred_system_register_access

Fields:

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

BRANCH_MISPREDICT

Simulating branch mispredict. Fields:

PC unsigned int
: Origin address (or 0 if unavailable)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

BRA_DIR

Direct branches, to immediate address. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR

Indirect branches, perhaps to a register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR_CCFAIL

Indirect branches fail, perhaps to a register.

BRA_RET_CCFAIL

Branche return fail, perhaps to a register.

CCFAIL

Conditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CCFAIL_UNC

Unconditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS

Conditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS_UNC

Unconditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CODE_CACHE_MAINT

Code cache maintenance. Fields:

END_INCL unsigned int
: Inclusive end addres for by VA or PA requests

NS bool
: Non-secure world for PA

START unsigned int
: Start address for by VA or PA requests

TYPE enum
: Request type.

COMPILE_BLOCK_END

Last instruction of basic block translated. Fields:

VADDR unsigned int
: Address of next instruction after this basic block

COMPILE_BLOCK_START

First instruction of basic block translated. Fields:

VADDR unsigned int
: Address of first instruction in basic block

COMPILE_INST

ARM instruction compiled. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DISASS string
: Disassembly of the instruction.

ISET enum
: The instruction set of this instruction.

ITSTATE unsigned int
: The ITSTATE current for the instruction.

OPCODE unsigned int
: The opcode of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

CONTEXTIDR

Context ID Register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

NS bool
: Secure or nonsecure banked register is accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

CONTEXT_SYNC

Called for every context synchronization event. Fields:

ARCH_CSE bool
: Architectural Context Synchronization Event

SYSREG_SYNC bool
: System register synchronization event

CORE_ENDIAN

Core BE8 Big-Endian state changed. Fields:

BE8 bool
: Core BE8 Big-Endian state

CORE_INFO

Static processor attributes. Only triggered by a call to DumpState(). Fields:

ARCH_PROFILE enum
: The architecture profile of the core.

CLUSTER_ID unsigned int
: The cluster ID of this processor.

CORE_NUM unsigned int
: The number of this core in an MP processor.

DETAILED_CORE_FEATURES enum
: Detailed core features as a bitfield. Enum values are bitmasks to extract the info from the field

FPU_VERSION enum
: The VFP version implemented by the core.

MEM_ARCH enum
: The memory architecture of the core.

MIDR_EL1 unsigned int
: MIDR_EL1 value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NUM_CORES unsigned int
: The number of cores in this MP processor.

QUANTUM_SIZE unsigned int
: The default quantum size of the core.

SECURITY_FEATURES bool
: Does the core have security features?

CORE_LOADS

Processor load accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CORE_REGS

Changes of the core registers R0 to R14. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ID unsigned int
: The register number, 0 to 14.

OLD_VALUE unsigned int
: The old value overwritten.

PHYS_ID enum
: The physical register accessed.

VALUE unsigned int
: The new value written to the register.

CORE_STORES

Processor store accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CRYPTO_SPEC

Every crypto instruction speculatively executed.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DATA_CACHE_SET_ALLOC_TAGS

Set allocation tags from DC GVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of memory in bytes.

TAG unsigned int
: The value of the tag being set.

VADDR unsigned int
: Virtual address used for setting allocation tags.

DATA_CACHE_ZERO

Zero and invalidate cache line from DC ZVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of zero'd memory in bytes.

VADDR unsigned int
: Virtual address of the zero'd cache line.

DEBUG_EVENT

Hardware debug support event. Fields:

EVENT enum
: Description of event

VALUE unsigned int
: data value

DWT_MATCH

DWT comparator matches. Fields:

NUM unsigned int
: DWT comparator number

TYPE enum
: DWT comparator configuration

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

END_COMPILE

Compilation end. Fields:

END_OF_PAGE_COUNT unsigned int
: Number of basic blocks exited due to page boundary since START_COMPILE.

FETCHFAIL_COUNT unsigned int
: Number of basic blocks exited due to ifetch failure START_COMPILE.

INST_COUNT unsigned int
: Number of instructions compiled since START_COMPILE.

NONSEQ_COUNT unsigned int
: Number of basic blocks exited due to non-sequential instructions since START_COMPILE.

PAGE_STRADDLE_COUNT unsigned int
: Number of basic blocks exited due to unaligned instructions crossing page since START_COMPILE.

EXCEPTION

Exceptions which are taken. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: The instruction set of the processor when the exception occurred.

LR unsigned int
: The value assigned to the link register.

NS enum
: The core's non-secure bit.

PC unsigned int
: The location where the exception occurred.

TARGET_ISET enum
: The instruction set of the exception handler code.

TARGET_PC unsigned int
: The address the exception branches to.

VECTOR enum
: The exception vector.

EXCEPTION_END

Every exception completed. Fields:

NS enum
: The core's non-secure bit.

EXCEPTION_ENTRY

Event marking the entry of an exception. All pushes of register to the stack will follow this event. Fields:

PC unsigned int
: The location where the exception occurred.

VECTOR enum
: The exception that occurred.

EXCEPTION_RAISE

Every exception raised.

EXCEPTION_RETURN

Marks the end of an exception, but on an M core does not indicate a program flow change (branch). In most situations it will be followed by an indirect branch. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

PC unsigned int
: The address of the branch instruction.

EXCEPTION_RETURN_PREBRANCH

Exception return event that must occur before PC and processor state have updated.

EXCEPTION_START

Every exception started. Fields:

NS enum
: The core's non-secure bit.

EXCEPTION_VECTOR_FETCH

I-side vector fetch for M-class exception. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Result of fetch if OK

RESPONSE enum
: 0=Aborted, 1=OK

VADDR unsigned int
: Where the vector is fetched from (including VTOR)

FP_STATE

Floating point state. Fields:

FPSR unsigned int
: FPSR register value.

MASK unsigned int
: The mask indicating updated cumulative bits.

HLT

HLT instruction occurred. Fields:

IMM16 unsigned int
: The 16-bit immediate value encoded in HLT instruction.

INFO_EXCEPTION_REASON

Provide information of real cause of exception. Fields:

FaultCause enum
: Which bit of which V7M/V8M-mainline FSR/HFSR would have been set.

PC unsigned int
: The location where the exception occurred.

PHASE enum
: What we are doing with the exception.

REASONS enum
: Why we are doing it.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SecurityState enum
: Privacy State of exception

VECTOR enum
: The exception that occurred.

INFO_FP_CONTEXT

Inform when ExecuteFPCheck writes CONTROL.FPCA and FPSCR etc. Fields:

WHAT enum
: What we are doing.

INFO_STACKING

Provide information of real cause of exception. Fields:

PHASE enum
: Are we starting or ending a sequence of stack writes.

REASONS enum
: Why we are doing it.

INST

Every instruction executed. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DEBUG_STATE bool
: The instruction is executed in debug state.

DISASS string
: Disassembly of instruction.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The current instruction set.

ITSTATE unsigned int
: The current ITSTATE.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

MODE enum
: The mode the core is in.

NS unsigned int
: The core's non-secure bit.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

OPCODE unsigned int
: The opcode of the instruction.

PADDR unsigned int
: The physical address of the instruction.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_END

Every instruction completed.

INST_START

Every instruction started. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

INST_STRADDLE

Instruction straddles boundary. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

ISB

ISB instruction occurred.

ITM

Instrumentation Trace Macrocell. Fields:

ITM_PACKET_TYPE enum
: ITM and DWT packets type.

PACKET_HEADER unsigned int
: ITM Packet Header.

PACKET_PAYLOAD unsigned int
: ITM Packet Payload.

LOAD_MULTIPLE_REGS_LIST

List of destination registers (not source) of a multiple registers load instruction. Fields:

REG string
: Register to add into the list

LOCAL_MONITOR

Local monitor activity. Fields:

PADDR unsigned int
: Local Monitor Address

State enum
: State of the monitor (Open/Exclusive)

LOCKUP_CYCLE

This event is triggered if the core enters or stays in the lock-up state. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CAUSE enum
: What causes the lockup?.

LOCKUP_ENTRY

This event is triggered if the core enters the lock-up state. Fields:

PC unsigned int
: The PC when the lockup state is entered.

LOCKUP_EXIT

This event is triggered if the core leaves the lock-up state.

MODE_CHANGE

Mode change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: The new mode.

OLD_MODE enum
: The old mode.

MPU_TRANS

Address translation information. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

EXEC_PERM enum
: Execution Permission

IDAU_REG_NUM enum
: IDAU region-number or code

INNERCACHE_RA bool
: Is the inner cache allocate on read

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

MEMTYPE enum
: Memory type.

NSDESC enum
: Security-attributes of the transaction.

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

PADDR unsigned int
: Address of the access.

PAGESIZE unsigned int
: Page size as log2(size).

READ_PERM enum
: Read Permission

REG_NUM signed int
: MPU region-number 0-15 (else 0xFF=none, 0xBB/0xDD=background/default)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SAU_REG_NUM enum
: SAU region-number or code

SAU_RGN_TYPE enum
: Privacy Region Type

SH enum
: Shareability

SIDE enum
: Inst / Data.

WRITE_PERM enum
: Write Permission

PERIODIC

Called for every quantum. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

PC unsigned int
: The address of the next instruction to be executed on this CPU.

PPB_READ_ACCESS

Trace reads from the core to memory mapped registers in the PPB (private peripheral bus) address range. Fields:

ACCESS_SIZE unsigned int
: Logarithm of the access size: 0=byte, 1=halfword, 2=word, 3=doubleword, ...

BURST_LEN unsigned int
: The number of data transfers (beats) in this burst.

DATA unsigned int
: The data transferred.

INNER_ATTRIBUTES unsigned int
: The inner attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

INST bool
: Is an instruction fetch.

LOCK enum
: Normal / Locked / Exclusive.

MASTER_ID unsigned int
: The AXI master ID.

NS bool
: Is a nonsecure access.

OUTER_ATTRIBUTES unsigned int
: The outer attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

PADDR unsigned int
: Physical address of access.

PRIV bool
: Is privileged access.

RESPONSE enum
: Whether the transaction was successful, or an error occurred.

USER_FLAGS unsigned int
: Core specific additional signals.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PPB_WRITE_ACCESS

Trace writes from the core to memory mapped registers in the PPB (private peripheral bus) address range. Fields:

ACCESS_SIZE unsigned int
: Logarithm of the access size: 0=byte, 1=halfword, 2=word, 3=doubleword, ...

BURST_LEN unsigned int
: The number of data transfers (beats) in this burst.

DATA unsigned int
: The data transferred.

INNER_ATTRIBUTES unsigned int
: The inner attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

INST bool
: Is an instruction fetch.

LOCK enum
: Normal / Locked / Exclusive.

MASTER_ID unsigned int
: The AXI master ID.

NS bool
: Is a nonsecure access.

OUTER_ATTRIBUTES unsigned int
: The outer attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

PADDR unsigned int
: Physical address of access.

PRIV bool
: Is privileged access.

RESPONSE enum
: Whether the transaction was successful, or an error occurred.

USER_FLAGS unsigned int
: Core specific additional signals.

PRELOAD_DATA

Data preload from PLD instruction. Fields:

VADDR unsigned int
: Virtual address of the data that should be preloaded.

PRELOAD_INST

Instruction preload from PLI instruction. Fields:

VADDR unsigned int
: Virtual address of the instruction that should be preloaded.

PRE_CORE_LOAD

Trace just before a core load. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_CORE_STORE

Trace just before a core store. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PSTATE_PACTIVE_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

PSTATE_PACTIVE_OPMODE_TRANSITION_DENIED

Trace the denied pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_POWER_TRANSITION

Trace pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

PSTATE_PACTIVE_POWER_TRANSITION_DENIED

Trace the denied pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_THREAD_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions for threads on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OLD_STATE enum
: Operating mode before the transition

RUN_STATE

Run state transition. Fields:

INST_COUNT unsigned int
: Ticks count at point of transition.

NEW enum
: New run state.

OLD enum
: Old run state.

SEMIHOSTING_CALL

Call of a semihost function occurred. Fields:

PC unsigned int
: The program counter after the semihosting call.

SEMIHOSTING_PRECALL

About to call semihost. Fields:

EL2 bool
: Use EL2 translation regime.

NS bool
: Is Non-Secure.

REG_WIDTH unsigned int
: The current register width in bytes.

SIGNAL

External signal state change. Fields:

SIGNAL enum
: Signal that changed

STATE bool
: Signal asserted state

START_COMPILE

Compilation started. Fields:

VADDR unsigned int
: Instruction where compilation begins.

STG_MULTIPLE_MICRO_OPS_ADDRESS_RANGE

Address range for MTE stores with multiple uops. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

STORE_MULTIPLE_REGS_ADDRESS_RANGE

Address range for stores with multiple registers. Fields:

ADDR_END unsigned int
: Final address.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR_START unsigned int
: Starting address.

SUBOPTIMAL_LDST_RETIRED

suboptimal load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1: half-word, 2: word ...)

OPTIMAL_ALIGN_SELECTOR unsigned int
: Selector to decide the optimal alignment size (1: access width, 2: 4-bytes, 3: 8-bytes, ..., 12: 4-kbytes)

VADDR unsigned int
: The virtual address of the access

SYNC

Called for every synchronization, that is every quantum break. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

LOCAL_QUANTUM unsigned int
: The local quantum of this CPU.

LOCAL_TIME unsigned int
: The local time of this CPU.

SYNC_REASON enum
: Reason for sync.

SYSCALL

System call instruction executed. Fields:

IMM unsigned int
: Immediate value of the system call instruction.

TYPE enum
: System call type.

VADDR unsigned int
: Instruction that caused the system call.

SYSREG_READ

Trace reads from system registers. Fields:

REG_NAME enum
: Register number

VALUE unsigned int
: The value read.

SYSREG_UPDATE32

Triggers when the system updates a register. Fields:

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SYSREG_UPDATE64

Triggers when the system updates a register. Fields:

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

SYSREG_WRITE

Trace write to system registers. Fields:

REG_ACCESS enum
: Register access status

REG_NAME enum
: Register number

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: Value written to the register.

UNALIGNED_LDST_RETIRED

Processor unaligned load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1=>half-word, 2->word ...)

NUMBER_OF_BEATS unsigned int
: Number of accesses (beats) in this burst (Total size in bytes is ACCESS_SIZE * NUMBER_OF_BEATS)

VADDR unsigned int
: The virtual address of the access

VFP_D_REGS

VFP/NEON D 64 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_Q_REGS

VFP/NEON Q 128 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding byte in the register. For example, 0x1 would indicate a write to VALUE[7:0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_SYS_REGS

Writes to the VFP/NEON units system registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: Which VFP system register is written.

MASK unsigned int
: The mask indicating updated cumulative bits.

OLD_VALUE unsigned int
: The register's old value overwritten.

VALUE unsigned int
: The new value written to the VFP system register.

VFP_S_REGS

VFP/NEON S 32 bit register write. Fields:

ALIASING enum
: Type of register aliasing used.

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

WAYPOINT

Signals end of basic block execution. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: Origin instructions set

IS_COND bool
: Indicates if this is a conditional waypoint.

PC unsigned int
: Origin address (or 0 if unavailable)

TAKEN bool
: Indicates if this waypoint was taken.

TARGET unsigned int
: Destination address

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

TARGET_ISET enum
Destination instructions set

WFE_END

WFE ended. Fields:

INST_COUNT unsigned int
Ticks count when leaving WFE.

WFE_EVENT_REGISTER

WFE event register status: set/clear, reason. Fields:

INST_COUNT unsigned int
Ticks count.
REASON enum
Reason for set/clear. Only REASON==0 (Cleared by WFE) clears the bit, all other reasons set it.

WFE_IGNORED

WFE ignored. Fields:

EVENT bool
This WFE was ignored because the event register was set.
INST_COUNT unsigned int
Ticks count when ignoring WFE.
TRAPPED bool
This WFE was trapped.

WFE_START

WFE entered. Fields:

INST_COUNT unsigned int
Ticks count when entering WFE.

WFI_END

WFI ended. Fields:

INST_COUNT unsigned int
Ticks count when leaving WFI.

WFI_IGNORED

WFI ignored. Fields:

DISABLED bool
This WFI was ignored because WFI is disabled.
INST_COUNT unsigned int
Ticks count when ignoring WFI.
REASON enum
specifies reason why WFI trace was ignored
TRAPPED bool
This WFI was trapped.

WFI_START

WFI entered. Fields:

INST_COUNT unsigned int
Ticks count when entering WFI.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

WFI_WAKEUP

WFI wakeup. Fields:

INST_COUNT unsigned int
: Ticks count when WFI wakeup occurred.

REASON enum
: Reason for wakeup.

XPSR

Changes to the xPSR register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

OLD_VALUE unsigned int
: The old xPSR value

UNKNOWN unsigned int
: Bits within the register that have unknown value.

VALUE unsigned int
: The new xPSR value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.35 ARM_Cortex-M4 - trace

This section describes the trace sources.

ASYNC_MEMORY_FAULT

Context ID Register write. Fields:

FAULT unsigned int
: Fault status

PADDR unsigned int
: Physical Address (or 0 if unavailable)

ATOMIC_END_ACCESS

Bus trace access for atomics. Fields:

ACCESS_FAIL bool
: Memory access failed

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

LOAD_VALUE unsigned int
: Loaded values

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ATOMIC_START_ACCESS

Bus trace access for atomics. Fields:

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ArchMsg.Error.exit_code

Fields:

EXITCODE signed int
: Exit code to be returned

KIND string
: Component kind that invoked the exit code trace

REASON string
: Reason why the exit code trace was invoked (optional)

ArchMsg.Warning.TarmacReconstructor

Unexpected event happened during Tarmac reconstruction. Fields:

REASON string
: Reason

ArchMsg.Warning.dap_csw_bad_size

A write to CM3DAP CSW has an invalid size field. Fields:

DATA unsigned int
: bits[2:0] are size

ArchMsg.Warning.recursive_branch

An instruction is performing a branch that targets the same instruction. If this was intended then a WFI might be more effective. Fields:

PC unsigned int
: Address of instruction causing the branch

ArchMsg.Warning.recursive_exception

An instruction has generated an exception that targets the same instruction. Fields:

CPSR unsigned int
: Processor state

PC unsigned int
: Address of instruction causing the exception

TYPE enum
: Type of exception

ArchMsg.Warning.secure_vector_fetch_from_nonsecure

Secure vector table is being fetched from Non-secure memory .

ArchMsg.Warning.sg_in_it_block

Arm recommends that software does not place SG (security gate instruction) inside an IT instruction block.

ArchMsg.warning_unpred_system_register_access

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

BRANCH_MISPREDICT

Simulating branch mispredict. Fields:

PC unsigned int
: Origin address (or 0 if unavailable)

BRA_DIR

Direct branches, to immediate address. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR

Indirect branches, perhaps to a register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR_CCFAIL

Indirect branches fail, perhaps to a register.

BRA_RET_CCFAIL

Branche return fail, perhaps to a register.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CCFAIL

Conditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCFAIL_UNC

Unconditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS

Conditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS_UNC

Unconditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CODE_CACHE_MAINT

Code cache maintenance. Fields:

END_INCL unsigned int
: Inclusive end addres for by VA or PA requests

NS bool
: Non-secure world for PA

START unsigned int
: Start address for by VA or PA requests

TYPE enum
: Request type.

COMPILE_BLOCK_END

Last instruction of basic block translated. Fields:

VADDR unsigned int
: Address of next instruction after this basic block

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

COMPILE_BLOCK_START

First instruction of basic block translated. Fields:

VADDR unsigned int
: Address of first instruction in basic block

COMPILE_INST

ARM instruction compiled. Fields:

DISASS string
: Disassembly of the instruction.

ISET enum
: The instruction set of this instruction.

ITSTATE unsigned int
: The ITSTATE current for the instruction.

OPCODE unsigned int
: The opcode of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

CONTEXTIDR

Context ID Register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

NS bool
: Secure or nonsecure banked register is accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

CONTEXT_SYNC

Called for every context synchronization event. Fields:

ARCH_CSE bool
: Architectural Context Synchronization Event

SYSREG_SYNC bool
: System register synchronization event

CORE_ENDIAN

Core BE8 Big-Endian state changed. Fields:

BE8 bool
: Core BE8 Big-Endian state

CORE_INFO

Static processor attributes. Only triggered by a call to DumpState(). Fields:

ARCH_PROFILE enum
: The architecture profile of the core.

CLUSTER_ID unsigned int
: The cluster ID of this processor.

CORE_NUM unsigned int
: The number of this core in an MP processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DETAILED_CORE_FEATURES enum
: Detailed core features as a bitfield. Enum values are bitmasks to extract the info from the field

FPU_VERSION enum
: The VFP version implemented by the core.

MEM_ARCH enum
: The memory architecture of the core.

MIDR_EL1 unsigned int
: MIDR_EL1 value

NUM_CORES unsigned int
: The number of cores in this MP processor.

QUANTUM_SIZE unsigned int
: The default quantum size of the core.

SECURITY_FEATURES bool
: Does the core have security features?

CORE_LOADS

Processor load accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VADDR unsigned int
: The virtual address of the access.

CORE_REGS

Changes of the core registers R0 to R14. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number, 0 to 14.

OLD_VALUE unsigned int
: The old value overwritten.

PHYS_ID enum
: The physical register accessed.

VALUE unsigned int
: The new value written to the register.

CORE_STORES

Processor store accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CRYPTO_SPEC

Every crypto instruction speculatively executed.

DATA_CACHE_SET_ALLOC_TAGS

Set allocation tags from DC GVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of memory in bytes.

TAG unsigned int
: The value of the tag being set.

VADDR unsigned int
: Virtual address used for setting allocation tags.

DATA_CACHE_ZERO

Zero and invalidate cache line from DC ZVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of zero'd memory in bytes.

VADDR unsigned int
: Virtual address of the zero'd cache line.

DEBUG_EVENT

Hardware debug support event. Fields:

EVENT enum
: Description of event

VALUE unsigned int
: data value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DWT_MATCH

DWT comparator matches. Fields:

NUM unsigned int
: DWT comparator number

TYPE enum
: DWT comparator configuration

END_COMPILE

Compilation end. Fields:

END_OF_PAGE_COUNT unsigned int
: Number of basic blocks exited due to page boundary since START_COMPILE.

FETCHFAIL_COUNT unsigned int
: Number of basic blocks exited due to ifetch failure START_COMPILE.

INST_COUNT unsigned int
: Number of instructions compiled since START_COMPILE.

NONSEQ_COUNT unsigned int
: Number of basic blocks exited due to non-sequential instructions since START_COMPILE.

PAGE_STRADDLE_COUNT unsigned int
: Number of basic blocks exited due to unaligned instructions crossing page since START_COMPILE.

EXCEPTION

Exceptions which are taken. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: The instruction set of the processor when the exception occurred.

LR unsigned int
: The value assigned to the link register.

PC unsigned int
: The location where the exception occurred.

TARGET_ISET enum
: The instruction set of the exception handler code.

TARGET_PC unsigned int
: The address the exception branches to.

VECTOR enum
: The exception vector.

EXCEPTION_END

Every exception completed.

EXCEPTION_ENTRY

Event marking the entry of an exception. All pushes of register to the stack will follow this event. Fields:

PC unsigned int
: The location where the exception occurred.

VECTOR enum
: The exception that occurred.

EXCEPTION_RAISE

Every exception raised.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

EXCEPTION_RETURN

Marks the end of an exception, but on an M core does not indicate a program flow change (branch). In most situations it will be followed by an indirect branch. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

PC unsigned int
: The address of the branch instruction.

EXCEPTION_RETURN_PREBRANCH

Exception return event that must occur before PC and processor state have updated.

EXCEPTION_START

Every exception started.

EXCEPTION_VECTOR_FETCH

I-side vector fetch for M-class exception. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Result of fetch if OK

RESPONSE enum
: 0=Aborted, 1=OK

VADDR unsigned int
: Where the vector is fetched from (including VTOR)

FP_STATE

Floating point state. Fields:

FPSR unsigned int
: FPSR register value.

MASK unsigned int
: The mask indicating updated cumulative bits.

HLT

HLT instruction occurred. Fields:

IMM16 unsigned int
: The 16-bit immediate value encoded in HLT instruction.

INFO_EXCEPTION_REASON

Provide information of real cause of exception. Fields:

FaultCause enum
: Which bit of which V7M/V8M-mainline FSR/HFSR would have been set.

PC unsigned int
: The location where the exception occurred.

PHASE enum
: What we are doing with the exception.

REASONS enum
: Why we are doing it.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VECTOR enum
: The exception that occurred.

INFO_FP_CONTEXT

Inform when ExecuteFPCheck writes CONTROL.FPCA and FPSCR etc. Fields:

WHAT enum
: What we are doing.

INFO_STACKING

Provide information of real cause of exception. Fields:

PHASE enum
: Are we starting or ending a sequence of stack writes.

REASONS enum
: Why we are doing it.

INST

Every instruction executed. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DEBUG_STATE bool
: The instruction is executed in debug state.

DISASS string
: Disassembly of instruction.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The current instruction set.

ITSTATE unsigned int
: The current ITSTATE.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

MODE enum
: The mode the core is in.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

OPCODE unsigned int
: The opcode of the instruction.

PADDR unsigned int
: The physical address of the instruction.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

INST_END

Every instruction completed.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_START

Every instruction started. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

INST_STRADDLE

Instruction straddles boundary. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

ISB

ISB instruction occurred.

ITM

Instrumentation Trace Macrocell. Fields:

ITM_PACKET_TYPE enum
: ITM and DWT packets type.

PACKET_HEADER unsigned int
: ITM Packet Header.

PACKET_PAYLOAD unsigned int
: ITM Packet Payload.

LOAD_MULTIPLE_REGS_LIST

List of destination registers (not source) of a multiple registers load instruction. Fields:

REG string
: Register to add into the list

LOCAL_MONITOR

Local monitor activity. Fields:

PADDR unsigned int
: Local Monitor Address

State enum
: State of the monitor (Open/Exclusive)

LOCKUP_CYCLE

This event is triggered if the core enters or stays in the lock-up state. Fields:

CAUSE enum
: What causes the lockup?.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOCKUP_ENTRY

This event is triggered if the core enters the lock-up state. Fields:

PC unsigned int
: The PC when the lockup state is entered.

LOCKUP_EXIT

This event is triggered if the core leaves the lock-up state.

MODE_CHANGE

Mode change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: The new mode.

OLD_MODE enum
: The old mode.

MPU_TRANS

Address translation information. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

EXEC_PERM enum
: Execution Permission

INNERCACHE_RA bool
: Is the inner cache allocate on read

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

MEMTYPE enum
: Memory type.

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

PADDR unsigned int
: Address of the access.

PAGESIZE unsigned int
: Page size as log2(size).

READ_PERM enum
: Read Permission

REG_NUM signed int
: MPU region-number 0-15 (else 0xFF=none, 0xBB/0xDD=background/default)

SH enum
: Shareability

SIDE enum
: Inst / Data.

WRITE_PERM enum
: Write Permission

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PERIODIC

Called for every quantum. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

PC unsigned int
: The address of the next instruction to be executed on this CPU.

PPB_READ_ACCESS

Trace reads from the core to memory mapped registers in the PPB (private peripheral bus) address range. Fields:

ACCESS_SIZE unsigned int
: Logarithm of the access size: 0=byte, 1=halfword, 2=word, 3=doubleword, ...

BURST_LEN unsigned int
: The number of data transfers (beats) in this burst.

DATA unsigned int
: The data transferred.

INNER_ATTRIBUTES unsigned int
: The inner attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

INST bool
: Is an instruction fetch.

LOCK enum
: Normal / Locked / Exclusive.

MASTER_ID unsigned int
: The AXI master ID.

NS bool
: Is a nonsecure access.

OUTER_ATTRIBUTES unsigned int
: The outer attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

PADDR unsigned int
: Physical address of access.

PRIV bool
: Is privileged access.

RESPONSE enum
: Whether the transaction was successful, or an error occurred.

USER_FLAGS unsigned int
: Core specific additional signals.

PPB_WRITE_ACCESS

Trace writes from the core to memory mapped registers in the PPB (private peripheral bus) address range. Fields:

ACCESS_SIZE unsigned int
: Logarithm of the access size: 0=byte, 1=halfword, 2=word, 3=doubleword, ...

BURST_LEN unsigned int
: The number of data transfers (beats) in this burst.

DATA unsigned int
: The data transferred.

INNER_ATTRIBUTES unsigned int
: The inner attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST bool
: Is an instruction fetch.

LOCK enum
: Normal / Locked / Exclusive.

MASTER_ID unsigned int
: The AXI master ID.

NS bool
: Is a nonsecure access.

OUTER_ATTRIBUTES unsigned int
: The outer attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

PADDR unsigned int
: Physical address of access.

PRIV bool
: Is privileged access.

RESPONSE enum
: Whether the transaction was successful, or an error occurred.

USER_FLAGS unsigned int
: Core specific additional signals.

PRELOAD_DATA

Data preload from PLD instruction. Fields:

VADDR unsigned int
: Virtual address of the data that should be preloaded.

PRELOAD_INST

Instruction preload from PLI instruction. Fields:

VADDR unsigned int
: Virtual address of the instruction that should be preloaded.

PRE_CORE_LOAD

Trace just before a core load. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_CORE_STORE

Trace just before a core store. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PSTATE_PACTIVE_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PSTATE_PACTIVE_OPMODE_TRANSITION_DENIED

Trace the denied pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_POWER_TRANSITION

Trace pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

PSTATE_PACTIVE_POWER_TRANSITION_DENIED

Trace the denied pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_THREAD_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions for threads on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

RUN_STATE

Run state transition. Fields:

INST_COUNT unsigned int
: Ticks count at point of transition.

NEW enum
: New run state.

OLD enum
: Old run state.

SEMIHOSTING_CALL

Call of a semihost function occurred. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PC unsigned int
: The program counter after the semihosting call.

SEMIHOSTING_PRECALL

About to call semihost. Fields:

EL2 bool
: Use EL2 translation regime.

NS bool
: Is Non-Secure.

REG_WIDTH unsigned int
: The current register width in bytes.

SIGNAL

External signal state change. Fields:

SIGNAL enum
: Signal that changed

STATE bool
: Signal asserted state

START_COMPILE

Compilation started. Fields:

VADDR unsigned int
: Instruction where compilation begins.

STG_MULTIPLE_MICRO_OPS_ADDRESS_RANGE

Address range for MTE stores with multiple uops. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

STORE_MULTIPLE_REGS_ADDRESS_RANGE

Address range for stores with multiple registers. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

SUBOPTIMAL_LDST_RETIRED

suboptimal load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1: half-word, 2: word ...)

OPTIMAL_ALIGN_SELECTOR unsigned int
: Selector to decide the optimal alignment size (1: access width, 2: 4-bytes, 3: 8-bytes, ..., 12: 4-kbytes)

VADDR unsigned int
: The virtual address of the access

SYNC

Called for every synchronization, that is every quantum break. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

LOCAL_QUANTUM unsigned int
: The local quantum of this CPU.

LOCAL_TIME unsigned int
: The local time of this CPU.

SYNC_REASON enum
: Reason for sync.

SYSCALL

System call instruction executed. Fields:

IMM unsigned int
: Immediate value of the system call instruction.

TYPE enum
: System call type.

VADDR unsigned int
: Instruction that caused the system call.

SYSREG_READ

Trace reads from system registers. Fields:

REG_NAME enum
: Register number

VALUE unsigned int
: The value read.

SYSREG_UPDATE32

Triggers when the system updates a register. Fields:

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

SYSREG_UPDATE64

Triggers when the system updates a register. Fields:

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

SYSREG_WRITE

Trace write to system registers. Fields:

REG_ACCESS enum
: Register access status

REG_NAME enum
: Register number

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: Value written to the register.

UNALIGNED_LDST_RETIRED

Processor unaligned load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1=>half-word, 2->word ...)

NUMBER_OF_BEATS unsigned int
: Number of accesses (beats) in this burst (Total size in bytes is ACCESS_SIZE * NUMBER_OF_BEATS)

VADDR unsigned int
: The virtual address of the access

VFP_D_REGS

VFP/NEON D 64 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_Q_REGS

VFP/NEON Q 128 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding byte in the register. For example, 0x1 would indicate a write to VALUE[7:0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_SYS_REGS

Writes to the VFP/NEON units system registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: Which VFP system register is written.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MASK unsigned int
: The mask indicating updated cumulative bits.

OLD_VALUE unsigned int
: The register's old value overwritten.

VALUE unsigned int
: The new value written to the VFP system register.

VFP_S_REGS

VFP/NEON S 32 bit register write. Fields:

ALIASING enum
: Type of register aliasing used.

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

WAYPOINT

Signals end of basic block execution. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: Origin instructions set

IS_COND bool
: Indicates if this is a conditional waypoint.

PC unsigned int
: Origin address (or 0 if unavailable)

TAKEN bool
: Indicates if this waypoint was taken.

TARGET unsigned int
: Destination address

TARGET_ISET enum
: Destination instructions set

WFE_END

WFE ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFE.

WFE_EVENT_REGISTER

WFE event register status: set/clear, reason. Fields:

INST_COUNT unsigned int
: Ticks count.

REASON enum
: Reason for set/clear. Only REASON==0 (Cleared by WFE) clears the bit, all other reasons set it.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

WFE_IGNORED

WFE ignored. Fields:

EVENT bool
: This WFE was ignored because the event register was set.

INST_COUNT unsigned int
: Ticks count when ignoring WFE.

TRAPPED bool
: This WFE was trapped.

WFE_START

WFE entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFE.

WFI_END

WFI ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFI.

WFI_IGNORED

WFI ignored. Fields:

DISABLED bool
: This WFI was ignored because WFI is disabled.

INST_COUNT unsigned int
: Ticks count when ignoring WFI.

REASON enum
: specifies reason why WFI trace was ignored

TRAPPED bool
: This WFI was trapped.

WFI_START

WFI entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFI.

WFI_WAKEUP

WFI wakeup. Fields:

INST_COUNT unsigned int
: Ticks count when WFI wakeup occurred.

REASON enum
: Reason for wakeup.

XPSR

Changes to the xPSR register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

OLD_VALUE unsigned int
: The old xPSR value

UNKNOWN unsigned int
: Bits within the register that have unknown value.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VALUE `unsigned int`
The new xPSR value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.36 ARM_Cortex-M55 - trace

This section describes the trace sources.

ASYNC_MEMORY_FAULT

Context ID Register write. Fields:

FAULT unsigned int
: Fault status

PADDR unsigned int
: Physical Address (or 0 if unavailable)

ATOMIC_END_ACCESS

Bus trace access for atomics. Fields:

ACCESS_FAIL bool
: Memory access failed

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

LOAD_VALUE unsigned int
: Loaded values

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ATOMIC_START_ACCESS

Bus trace access for atomics. Fields:

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ArchMsg.Error.exit_code

Fields:

EXITCODE signed int
: Exit code to be returned

KIND string
: Component kind that invoked the exit code trace

REASON string
: Reason why the exit code trace was invoked (optional)

ArchMsg.Warning.TarmacReconstructor

Unexpected event happened during Tarmac reconstruction. Fields:

REASON string
: Reason

ArchMsg.Warning.Unpredictably Indexed PM Event Register

Fields:

IsDirect unsigned int
: Direct Access rather than indirect

N unsigned int
: Number of Accessible Registers

SEL unsigned int
: Selector

ArchMsg.Warning.branch_to_unaligned_address

Fields:

ADDR unsigned int
: Unaligned branch address

ArchMsg.Warning.cache_contents_unknown

A cache has been activated without being invalidated and may contain UNKNOWN content. Fields:

REGIME unsigned int
: Current translation regime

SIDE unsigned int
: 0 d-side cache or unified cache, 1 i-side

ArchMsg.Warning.dap_csw_bad_size

A write to CM3DAP CSW has an invalid size field. Fields:

DATA unsigned int
: bits[2:0] are size

ArchMsg.Warning.dcimvac_matches_watchpoint

Watchpoints matching an AArch32 DCIMVA are implementation defined.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_fieldmismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_fracbitsnegative

VCVT Instruction is unpredictable with a negative number of fraction bits.

ArchMsg.Warning.decode_initblock

Instruction is unpredictable in an IT block.

ArchMsg.Warning.decode_initblocknotlast

Instruction is unpredictable in an IT block when not the last.

ArchMsg.Warning.decode_invalid_condition

Instruction is unpredictable when the condition is not AL.

ArchMsg.Warning.decode_invalidfieldcombination

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_invalidvalue

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_mem_hint_unallocated

There is no memory hint allocated to this bit pattern.

ArchMsg.Warning.decode_registermatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registermismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registeroutofrange

Float point instruction is unpredictable indexing beyond the end of the register bank.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_sbzsbo

Reserved bits in the instruction are not canonical.

ArchMsg.Warning.decode_transactiontoobig

Attempt to perform a memory transaction of more than 128 bytes.

ArchMsg.Warning.decode_unpred_other

Instruction is unpredictable.

ArchMsg.Warning.decode_unpreduseofpc

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13inreglist

Instruction is unpredictable with R13 in register-list.

ArchMsg.Warning.decode_useofd16orhigher

Attempt to use register D16 or higher where it is not allowed.

ArchMsg.Warning.decode_writebackandbaseinlist

Instruction is unpredictable with write back when the base register is in the transfer list.

ArchMsg.Warning.decode_zeroregistersinlist

Instruction is unpredictable when no registers are to be transfered.

ArchMsg.Warning.extension_unpred

Probably unpred in IT state or decod. Fields:

PC unsigned int
: Address of this instruction

ArchMsg.Warning.msr_zeroes_ltpsize

May be pre-MVE setup code.

ArchMsg.Warning.pmu_and_dwt

Fields:

IS_PMU bool
: Whether trying to access the PMU or the DWT

ArchMsg.Warning.recursive_branch

An instruction is performing a branch that targets the same instruction. If this was intended then a WFI might be more effective. Fields:

PC unsigned int
: Address of instruction causing the branch

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.recursive_exception

An instruction has generated an exception that targets the same instruction. Fields:

CPSR unsigned int
: Processor state

PC unsigned int
: Address of instruction causing the exception

TYPE enum
: Type of exception

ArchMsg.Warning.reentrant_vector_catch

Fields:

ADDR unsigned int
: vector address

ArchMsg.Warning.reserved_it_state

Fields:

ITSTATE unsigned int
: reserved IT state

ArchMsg.Warning.secure_vector_fetch_from_nonsecure

Secure vector table is being fetched from Non-secure memory .

ArchMsg.Warning.sg_in_it_block

Arm recommends that software does not place SG (security gate instruction) inside an IT instruction block.

ArchMsg.Warning.unpredictable_a32_breakpoint

Fields:

ADDR1 unsigned int
: Address

ADDR2 unsigned int
: Breakpoint address

BAS unsigned int
: BAS Field

IS_ADDRESS_MISMATCH_BREAKPOINT bool
: Is Mismatch breakpoint, not a match breakpoint

N unsigned int
: Breakpoint number

OFFSET unsigned int
: Indicates which halfword actually considering into the A32 instruction

ArchMsg.Warning.unpredictable_banked_register_access

Fields:

R unsigned int
: R bit

SYSM unsigned int
: SYSm

ArchMsg.Warning.unpredictable_ccfail_undef

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPCODE unsigned int
opcode of the instruction
VADDR unsigned int
Virtual address of the instruction being executed

ArchMsg.Warning.unpredictable_ici_bits

ICI bits do not apply or are architecturally unpredictable for instruction.

ArchMsg.Warning.unpredictable_t32_breakpoint

Fields:

ADDR1 unsigned int
Address
ADDR2 unsigned int
Breakpoint address
BAS unsigned int
BAS Field
IS_ADDRESS_MISMATCH_BREAKPOINT bool
Is Mismatch breakpoint, not a match breakpoint
N unsigned int
Breakpoint number
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_t32_vector_catch

Fields:

ADDR unsigned int
Address
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_watchpoint_far

Fields:

LOWER_BOUND unsigned int
the lowest address accessed by the instruction that triggered the watchpoint
UPPER_BOUND unsigned int
the highest watchpointed address accessed by that instruction

ArchMsg.Warning.warning_AdvSIMDExpandImmUnexpectedZero

AdvSIMDExpandImm may treat this immediate value as UNPREDICTABLE.

ArchMsg.Warning.warning_ConditionalSMC

SMC instruction has UNPREDICTABLE effects when conditional when combined with traps.

ArchMsg.Warning.warning_access_crosses_page_boundary

Access crosses page boundary.

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault_on_both_pages

Access crosses page boundary has fault(s).

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_access_crosses_page_boundary_spanning_different_memory

Fields:

ADDR unsigned int
: address of access crossing page

MEMTYPE_PAGE1 signed int
: First Page's memory type (0-Normal, 1-Non-Normal)

MEMTYPE_PAGE2 signed int
: Second Page's memory type (0-Normal, 1-Non-Normal)

ArchMsg.Warning.warning_access_wraps_around_memory

Fields:

WIDTH unsigned int
: Address width

ArchMsg.Warning.warning_bx_from_thumbee

Fields:

ADDR unsigned int
: Destination address

ArchMsg.Warning.warning_ccsidr_unimplemented_level

Fields:

CSSELR unsigned int
: current effective value of CSSELR

ArchMsg.Warning.warning_change_to_ns_when_tge_set

Attempting to change NS to 1 when HCR.TGE = 1.

ArchMsg.Warning.warning_contiguous_bit_check_abort

Fields:

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ArchMsg.Warning.warning_contiguous_bit_error

Fields:

CONTIG_ADDR unsigned int
: address of the TLB first read, with which this entry is expected to be contiguous

CONTIG_DATA unsigned int
: expected contents based on the entry at CONTIG_ADDR

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ENTRY_DATA unsigned int
: contents of conflicting TLB entry

ArchMsg.Warning.warning_cp10_cp11_mismatch

Fields:

EL unsigned int
: EL owning the control

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_cp10_cp11_reserved_value

Fields:

CP10 unsigned int
: CP10 access

CP11 unsigned int
: CP11 access

ArchMsg.Warning.warning_csselr_level_out_of_range

Fields:

IMPLEMENTED unsigned int
: number of cache levels implemented

LEVEL unsigned int
: written value of CSSELR.Level

TYPE enum
: The type of cache selected by CSSELR.InD and TnD

ArchMsg.Warning.warning_debug_flow_control_bits_not_obeyed

Fields:

ERROR enum
: type of error resulting

REGISTER enum
: register under consideration

ArchMsg.Warning.warning_debug_register_access_during_reset

Fields:

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

ArchMsg.Warning.warning_decode_cps_inconsistent_fields

The mode setting fields in the CPS instruction are inconsistent.

ArchMsg.Warning.warning_decode_invalid_state

Instruction is unpredictable in current exception-level/security state.

ArchMsg.Warning.warning_default_cacheable_mmu_on

Default cacheable is enabled (HCR.DC = 1) while MMU is enabled.

ArchMsg.Warning.warning_default_cacheable_vmmu_off

Default cacheable is enabled (HCR.DC = 1) while HCR.VM = 0.

ArchMsg.Warning.warning_deprecated_wvr_bit_2_set

Fields:

N unsigned int
: index

WVR unsigned int
: DBGWVR

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_eret_while_software_step_active_pending

Missing ISB between setting MDSCR_EL1.SS and ERET.

ArchMsg.Warning.warning_exclusive_to_non_normal

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_exclusive_to_non_writeback

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_execute_from_device_memory

An attempt was made to execute from Device memory.

ArchMsg.Warning.warning_illegal_cpsr_mode

Fields:

NEW_MODE unsigned int
: New value of CPSR.M

ArchMsg.Warning.warning_illegal_srs_mode

Fields:

MODE unsigned int
: mode for Banked SP

ArchMsg.Warning.warning_implementation_defined_read_debug_register_in_SCS

Read Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_implementation_defined_sequential_security_transitions_supported

The behaviour of a sequential instruction fetches that cross from non-secure to secure memory and contain SG instruction is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_implementation_defined_stack_limit_check_supported

It is IMPLEMENTATION DEFINED whether stack pointer limit checking is performed for the this instructions. Fields:

ADDRESS unsigned int
: Address of instruction

OPCODE unsigned int
: instruction opcode

ArchMsg.Warning.warning_implementation_defined_write_debug_register_in_SCS

Write Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

DATA unsigned int
: data attempted to be written

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PPB_OFFSET unsigned int
debug register offset

ArchMsg.Warning.warning_invalid_tcr_granule

Fields:

REQUEST signed int
page size requested (or 0 for reserved)
SUBSTITUTE signed int
best guess available page size
TG_ID bool
bits TG0 or TG1

ArchMsg.Warning.warning_load_multiple_user_registers_from_user_mode

An LDM(user registers) instruction executed from user mode.

ArchMsg.Warning.warning_load_pc_from_unaligned

PC loaded from an unaligned location.

ArchMsg.Warning.warning_mte_in_mem_attr_not_supported

Fields:

ATTR unsigned int
Memory attribute

ArchMsg.Warning.warning_reserved_breakpoint_state_match

Fields:

IS_BREAKPOINT unsigned int
Is breakpoint, not watchpoint

ArchMsg.Warning.warning_shareability

Unpredictable: combination of the 1st and 2nd stages of translation is Normal Inner Non-Cacheable, Outer Non-Cacheable.

ArchMsg.Warning.warning_software_step_set_while_enabled

MDSCR_EL1.SS set to 1 while software step debug exceptions are enabled.

ArchMsg.Warning.warning_thumb_instruction_wraps_around_memory

Thumb instruction warps around memory.

ArchMsg.Warning.warning_ttbr_sbz_bits_are_not_zero

Fields:

MSB unsigned int
x-1
TTBR enum
which TTBR register

ArchMsg.Warning.warning_unaligned_address_dbgdtrrx_write

Fields:

ADDR unsigned int
Address

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unaligned_address_dbgdtrtx_read

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_unaligned_to_device

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_unaligned_to_strongly_ordered

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_unknown_sau_rnr

SAU_RNR was set to an unsupported value.

ArchMsg.Warning.warning_unpredictable_AIRCR_PRIP_and_BFHFNMINP

The effect of setting both AIRCR.BFHFNMINP and AIRCR.PRIP to 1 is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTCLRACTIVE_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTCLRACTIVE if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_and_SYSRESETREQ

When the processor is halted in Debug state, if a write to the register writes a 1 to both VECTRESET and SYSRESETREQ, the behavior is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTRESET if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_incorrect_VKEY

The value 0x05FA must be written to AIRCR.VKEY, otherwise the register write is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_EXE_RETURN_Reserved_Bit

EXC_RETURN[23:7] are reserved with the special condition that all bits should be written as one. Values other than all 1s are UNPREDICTABLE. Fields:

EXC_RETURN unsigned int
: Special PC value

SBO_MASK unsigned int
: Which bits Should Be One to not UNPRED

ArchMsg.Warning.warning_unpredictable_change_to_DEMCR_MON_STEP_at_insufficient_priority

The effect of changing DEMCR.MON_STEP at an execution priority that is lower than the priority of the DebugMonitor exception is CONSTRAINED UNPREDICTABLE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_change_to_priority_of_active_exception

Changing the priority of an active exception is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_clear_lspact

Fields:

DRVEXC_TAKEN unsigned int
: derived exception taken not pended

REACHED_LAST_STORE unsigned int
: only the last faulted

ArchMsg.Warning.warning_unpredictable_dbg_exit_unaligned_dlr

Debug exit to AArch32 with DLR[0] set to 1. PC[0] can be set to 0 or to DLR[0].

ArchMsg.Warning.warning_unpredictable_exception_catch

Generation of an exception catch event is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_inconsistent_state

state on exception return is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_instruction

exception return instruction is unpredictable.

ArchMsg.Warning.warning_unpredictable_in_debug_state

Instruction is UNPREDICTABLE when executed in debug state.

ArchMsg.Warning.warning_unpredictable_mair_encoding

Fields:

ATTR unsigned int
: Memory attribute

ArchMsg.Warning.warning_unpredictable_pmu_counter_access

Access to PMU counters from non-secure EL0 or EL1 is UNPREDICTABLE with MDCR_EL2.HPMN set to 0.

ArchMsg.Warning.warning_unpredictable_prioritization_breakpoint_match_vector_catch

Two events with the same priority occurred at the same instruction: Address Matching Vector Catch debug event, and Breakpoint debug event It is constrained unpredictable which is taken.

ArchMsg.Warning.warning_unpredictable_stack_selection

UNPREDICTABLE state on context switch.

ArchMsg.Warning.warning_unpredictable_tsize_out_of_range

Fields:

TCR enum
: Which TCR

TSIZE signed int
: Value of TSize

TSIZE_MAX signed int
: The maximum allowed value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

TTBR unsigned int
: Which TTBR region

ArchMsg.Warning.warning_unpredictable_unaligned_pc_as_base_register

The PC value must be word-aligned, otherwise the behavior of the instruction is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_unaligned_pop_stack

Fields:

FRAME_ADDR unsigned int
: unaligned stack frame address

ArchMsg.Warning.warning_unpredictable_vmsa_memattrib

Fields:

TEXCB unsigned int
: texcb value

TEX_REMAP bool
: use tex remap

ArchMsg.Warning.warning_unpredictable_vtcr_t0sz_sl0

Fields:

SL0 signed int
: VTCR.SL0, starting level for stage 2 translation table walks

T0SZ signed int
: VTCR.T0SZ, required input address range

ArchMsg.Warning.warning_unpredictable_write_DHCSR

Fields:

NEW unsigned int
: data attempted to be written

OLD unsigned int
: previous value

ArchMsg.Warning.warning_unpredictable_write_SHCSR

UNPREDICTABLE setting a pending bit to 1 for an exception with priority >= execution priority.

ArchMsg.Warning.warning_unsupported_access_to_memory_mapped_register

Access to memory mapped register is unsupported.

ArchMsg.Warning.warning_user_jmcr_access

Fields:

WRITE unsigned int
: access is write

ArchMsg.Warning.warning_wcr_mask_and_bas

DBGWCR_EL1.MASK is non-zero and BAS is not 0xff.

ArchMsg.Warning.warning_wcr_mask_reserved

DBGWCR_ELn.MASK is set to a reserved value.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_wcr_non_configuous_bas

DBGWCR_EL1.BAS has non-contiguous set of ones.

ArchMsg.Warning.warning_wvr_masked_bit_not_zero

DBGWVRn_EL1.VA contains a bit masked by DBGWCRn_EL1.MASK that is not zero.

ArchMsg.warning_non_architectural_beat_order

Current set of commited beats does not give legal ECI value.

ArchMsg.warning_unpred_system_register_access

Fields:

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

BEAT_EXCEPTION

Exception during of one beat of an M. Fields:

BEAT unsigned int
: Beat number 0..3

EXC unsigned int
: Internal representation of the excep

EXCEPTION_PC unsigned int
: Address of older in-flight instructi

INST unsigned int
: This instruction as FirstHalfWord:Se

INST_ADDR unsigned int
: Address of faulting instruction inst

BEAT_START

Start of one beat of an MVE Vector i. Fields:

BEAT unsigned int
: Beat number 0..3

DISASS string
: Disassembly of instruction.

ELMT_MASK unsigned int
: Element Mask

EXCEPTION_PC unsigned int
: Address of older in-flight instructi

INST unsigned int
: This instruction as FirstHalfWord:Se

INST_ADDR unsigned int
: Address of this instruction

LTP_MASK unsigned int
: Element Mask from Loop Tail Predicat

VPT_MASK unsigned int
: Element Mask from VPR

BRANCH_MISPREDICT

Simulating branch mispredict. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PC unsigned int
: Origin address (or 0 if unavailable)

BRA_DIR

Direct branches, to immediate address. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR

Indirect branches, perhaps to a register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR_CCFAIL

Indirect branches fail, perhaps to a register.

BRA_LOB

Low Overhead branch. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_RET_CCFAIL

Branche return fail, perhaps to a register.

CACHE_MAINTENANCE_OP

Cache Maintenance Operation. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Specified MVA or Set/way

FUNCTION enum
: Clean / Invalidate

SCOPE enum
: Affected region

SIDE enum
: Inst / Data.

CCFAIL

Conditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCFAIL_UNC

Unconditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS

Conditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PC unsigned int
: The address of the conditional instruction.

CCPASS_UNC

Unconditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CODE_CACHE_MAINT

Code cache maintenance. Fields:

END_INCL unsigned int
: Inclusive end addres for by VA or PA requests

NS bool
: Non-secure world for PA

START unsigned int
: Start address for by VA or PA requests

TYPE enum
: Request type.

COMPILE_BLOCK_END

Last instruction of basic block translated. Fields:

VADDR unsigned int
: Address of next instruction after this basic block

COMPILE_BLOCK_START

First instruction of basic block translated. Fields:

VADDR unsigned int
: Address of first instruction in basic block

COMPILE_INST

ARM instruction compiled. Fields:

DISASS string
: Disassembly of the instruction.

ISET enum
: The instruction set of this instruction.

ITSTATE unsigned int
: The ITSTATE current for the instruction.

OPCODE unsigned int
: The opcode of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

CONTEXTIDR

Context ID Register write. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

NS bool
: Secure or nonsecure banked register is accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

CONTEXT_SYNC

Called for every context synchronization event. Fields:

ARCH_CSE bool
: Architectural Context Synchronization Event

SYSREG_SYNC bool
: System register synchronization event

CORE_ENDIAN

Core BE8 Big-Endian state changed. Fields:

BE8 bool
: Core BE8 Big-Endian state

CORE_INFO

Static processor attributes. Only triggered by a call to DumpState(). Fields:

ARCH_PROFILE enum
: The architecture profile of the core.

CLUSTER_ID unsigned int
: The cluster ID of this processor.

CORE_NUM unsigned int
: The number of this core in an MP processor.

DETAILED_CORE_FEATURES enum
: Detailed core features as a bitfield. Enum values are bitmasks to extract the info from the field

FPU_VERSION enum
: The VFP version implemented by the core.

MEM_ARCH enum
: The memory architecture of the core.

MIDR_EL1 unsigned int
: MIDR_EL1 value

NUM_CORES unsigned int
: The number of cores in this MP processor.

QUANTUM_SIZE unsigned int
: The default quantum size of the core.

SECURITY_FEATURES bool
: Does the core have security features?

CORE_LOADS

Processor load accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CORE_REGS

Changes of the core registers R0 to R14. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number, 0 to 14.

OLD_VALUE unsigned int
: The old value overwritten.

PHYS_ID enum
: The physical register accessed.

VALUE unsigned int
: The new value written to the register.

CORE_STORES

Processor store accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CRYPTO_SPEC

Every crypto instruction speculatively executed.

DATA_CACHE_SET_ALLOC_TAGS

Set allocation tags from DC GVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SIZE unsigned int
: The size of memory in bytes.

TAG unsigned int
: The value of the tag being set.

VADDR unsigned int
: Virtual address used for setting allocation tags.

DATA_CACHE_ZERO

Zero and invalidate cache line from DC ZVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of zero'd memory in bytes.

VADDR unsigned int
: Virtual address of the zero'd cache line.

DEBUG_EVENT

Hardware debug support event. Fields:

EVENT enum
: Description of event

VALUE unsigned int
: data value

DTCM_ACCESS

Trace TCM accesses.

DTCM_ACCESS_RAZ_WI

Trace TCM accesses. Fields:

ADDR unsigned int
: Address of the RAZ/WI access.

DWT_MATCH

DWT comparator matches. Fields:

NUM unsigned int
: DWT comparator number

TYPE enum
: DWT comparator configuration

END_COMPILE

Compilation end. Fields:

END_OF_PAGE_COUNT unsigned int
: Number of basic blocks exited due to page boundary since START_COMPILE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

FETCHFAIL_COUNT unsigned int
: Number of basic blocks exited due to ifetch failure START_COMPILE.

INST_COUNT unsigned int
: Number of instructions compiled since START_COMPILE.

NONSEQ_COUNT unsigned int
: Number of basic blocks exited due to non-sequential instructions since START_COMPILE.

PAGE_STRADDLE_COUNT unsigned int
: Number of basic blocks exited due to unaligned instructions crossing page since START_COMPILE.

EXCEPTION

Exceptions which are taken. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: The instruction set of the processor when the exception occurred.

LR unsigned int
: The value assigned to the link register.

NS enum
: The core's non-secure bit.

PC unsigned int
: The location where the exception occurred.

TARGET_ISET enum
: The instruction set of the exception handler code.

TARGET_PC unsigned int
: The address the exception branches to.

VECTOR enum
: The exception vector.

EXCEPTION_END

Every exception completed. Fields:

NS enum
: The core's non-secure bit.

EXCEPTION_ENTRY

Event marking the entry of an exception. All pushes of register to the stack will follow this event. Fields:

PC unsigned int
: The location where the exception occurred.

VECTOR enum
: The exception that occurred.

EXCEPTION_RAISE

Every exception raised.

EXCEPTION_RETURN

Marks the end of an exception, but on an M core does not indicate a program flow change (branch). In most situations it will be followed by an indirect branch. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ISET enum
: The instructions set of the branch instruction.

PC unsigned int
: The address of the branch instruction.

EXCEPTION_RETURN_PREBRANCH

Exception return event that must occur before PC and processor state have updated.

EXCEPTION_START

Every exception started. Fields:

NS enum
: The core's non-secure bit.

EXCEPTION_VECTOR_FETCH

I-side vector fetch for M-class exception. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Result of fetch if OK

RESPONSE enum
: 0=Aborted, 1=OK

VADDR unsigned int
: Where the vector is fetched from (including VTOR)

FP_STATE

Floating point state. Fields:

FPSR unsigned int
: FPSR register value.

MASK unsigned int
: The mask indicating updated cumulative bits.

HLT

HLT instruction occurred. Fields:

IMM16 unsigned int
: The 16-bit immediate value encoded in HLT instruction.

INFO_EXCEPTION_REASON

Provide information of real cause of exception. Fields:

FaultCause enum
: Which bit of which V7M/V8M-mainline FSR/HFSR would have been set.

PC unsigned int
: The location where the exception occurred.

PHASE enum
: What we are doing with the exception.

REASONS enum
: Why we are doing it.

SecurityState enum
: Privacy State of exception

VECTOR enum
: The exception that occurred.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INFO_FP_CONTEXT

Inform when ExecuteFPCheck writes CONTROL.FPCA and FPSCR etc. Fields:

WHAT enum
: What we are doing.

INFO_STACKING

Provide information of real cause of exception. Fields:

PHASE enum
: Are we starting or ending a sequence of stack writes.

REASONS enum
: Why we are doing it.

INST

Every instruction executed. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DEBUG_STATE bool
: The instruction is executed in debug state.

DISASS string
: Disassembly of instruction.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The current instruction set.

ITSTATE unsigned int
: The current ITSTATE.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

MODE enum
: The mode the core is in.

NS unsigned int
: The core's non-secure bit.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

OPCODE unsigned int
: The opcode of the instruction.

PADDR unsigned int
: The physical address of the instruction.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

INST_END

Every instruction completed.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_START

Every instruction started. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

INST_STRADDLE

Instruction straddles boundary. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

ISB

ISB instruction occurred.

ITCM_ACCESS

Trace TCM accesses.

ITCM_ACCESS_RAZ_WI

Trace TCM accesses. Fields:

ADDR unsigned int
: Address of the RAZ/WI access.

ITM

Instrumentation Trace Macrocell. Fields:

ITM_PACKET_TYPE enum
: ITM and DWT packets type.

PACKET_HEADER unsigned int
: ITM Packet Header.

PACKET_PAYLOAD unsigned int
: ITM Packet Payload.

LOAD_MULTIPLE_REGS_LIST

List of destination registers (not source) of a multiple registers load instruction. Fields:

REG string
: Register to add into the list

LOCAL_MONITOR

Local monitor activity. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PADDR unsigned int
: Local Monitor Address

State enum
: State of the monitor (Open/Exclusive)

LOCKUP_CYCLE

This event is triggered if the core enters or stays in the lock-up state. Fields:

CAUSE enum
: What causes the lockup?.

LOCKUP_ENTRY

This event is triggered if the core enters the lock-up state. Fields:

PC unsigned int
: The PC when the lockup state is entered.

LOCKUP_EXIT

This event is triggered if the core leaves the lock-up state.

LO_BRANCH_INFO

Low Overhead Branch cache Info. Fields:

BF unsigned int
: Value originates from a BF instructi

CAUSE enum
: What caused the update

END_ADDR unsigned int
: Address of the instruction after the

JUMP_ADDR unsigned int
: Jump address

LF unsigned int
: Link/forever

T16IND unsigned int
: T16 indirect branch

VALID unsigned int
: Entry is valid

LO_BRANCH_INSTR

Low Overhead Branch Instruction. Fields:

EFFECT enum
: Did it set LO_BRANCH_INFO and/or branch

INSTR enum
: What sort of instruction (BF/../BFCSEL/LE/WLS/...)

MODE_CHANGE

Mode change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: The new mode.

OLD_MODE enum
: The old mode.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MPU_TRANS

Address translation information. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

EXEC_PERM enum
: Execution Permission

IDAU_REG_NUM enum
: IDAU region-number or code

INNERCACHE_RA bool
: Is the inner cache allocate on read

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

MEMTYPE enum
: Memory type.

NSDESC enum
: Security-attributes of the transaction.

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

PADDR unsigned int
: Address of the access.

PAGESIZE unsigned int
: Page size as log2(size).

READ_PERM enum
: Read Permission

REG_NUM signed int
: MPU region-number 0-15 (else 0xFF=none, 0xBB/0xDD=background/default)

SAU_REG_NUM enum
: SAU region-number or code

SAU_RGN_TYPE enum
: Privacy Region Type

SH enum
: Shareability

SIDE enum
: Inst / Data.

WRITE_PERM enum
: Write Permission

PERIODIC

Called for every quantum. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

PC unsigned int
: The address of the next instruction to be executed on this CPU.

PMU_COUNTER_OVERFLOW

PMU counter overflow. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

EVENT_ID unsigned int
: Event Identifier

INDEX unsigned int
: Counter index (as selected by PMSELR).

INTERRUPT bool
: Is interrupt enabled on overflow for this counter

PMU_M_EVENT_BF_CANCEL

PMU Event BF_CANCEL.

PMU_M_EVENT_BF_RETIRED

PMU Event BF_RETIRED.

PMU_M_EVENT_BF_SPEC

PMU Event BF_SPEC.

PMU_M_EVENT_DWT_CMPMATCH0

PMU Event DWT_CMPMATCH0.

PMU_M_EVENT_DWT_CMPMATCH1

PMU Event DWT_CMPMATCH1.

PMU_M_EVENT_DWT_CMPMATCH2

PMU Event DWT_CMPMATCH2.

PMU_M_EVENT_DWT_CMPMATCH3

PMU Event DWT_CMPMATCH3.

PMU_M_EVENT_INST_SPEC

PMU Event INST_SPEC.

PMU_M_EVENT_LE_CANCEL

PMU Event LE_CANCEL.

PMU_M_EVENT_LE_RETIRED

PMU Event LE_RETIRED.

PMU_M_EVENT_LE_SPEC

PMU Event LE_SPEC.

PMU_M_EVENT_MVE_FP_HP_RETIRED

PMU Event MVE_FP_HP_RETIRED.

PMU_M_EVENT_MVE_FP_HP_SPEC

PMU Event MVE_FP_HP_SPEC.

PMU_M_EVENT_MVE_FP_MAC_RETIRED

PMU Event MVE_FP_MAC_RETIRED.

PMU_M_EVENT_MVE_FP_MAC_SPEC

PMU Event MVE_FP_MAC_SPEC.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PMU_M_EVENT_MVE_FP_RETIRED

PMU Event MVE_FP_RETIRED.

PMU_M_EVENT_MVE_FP_SPEC

PMU Event MVE_FP_SPEC.

PMU_M_EVENT_MVE_FP_SP_RETIRED

PMU Event MVE_FP_SP_RETIRED.

PMU_M_EVENT_MVE_FP_SP_SPEC

PMU Event MVE_FP_SP_SPEC.

PMU_M_EVENT_MVE_INST_RETIRED

PMU Event MVE_INST_RETIRED.

PMU_M_EVENT_MVE_INST_SPEC

PMU Event MVE_INST_SPEC.

PMU_M_EVENT_MVE_INT_MAC_RETIRED

PMU Event MVE_INT_MAC_RETIRED.

PMU_M_EVENT_MVE_INT_MAC_SPEC

PMU Event MVE_INT_MAC_SPEC.

PMU_M_EVENT_MVE_INT_RETIRED

PMU Event MVE_INT_RETIRED.

PMU_M_EVENT_MVE_INT_SPEC

PMU Event MVE_INT_SPEC.

PMU_M_EVENT_MVE_LDST_MULTI_RETIRED

PMU Event MVE_LDST_MULTI_RETIRED.

PMU_M_EVENT_MVE_LDST_MULTI_SPEC

PMU Event MVE_LDST_MULTI_SPEC.

PMU_M_EVENT_MVE_LDST_RETIRED

PMU Event MVE_LDST_RETIRED.

PMU_M_EVENT_MVE_LDST_SPEC

PMU Event MVE_LDST_SPEC.

PMU_M_EVENT_MVE_LD_MULTI_RETIRED

PMU Event MVE_LD_MULTI_RETIRED.

PMU_M_EVENT_MVE_LD_MULTI_SPEC

PMU Event MVE_LD_MULTI_SPEC.

PMU_M_EVENT_MVE_LD_RETIRED

PMU Event MVE_LD_RETIRED.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PMU_M_EVENT_MVE_LD_SPEC

PMU Event MVE_LD_SPEC.

PMU_M_EVENT_MVE_PRED

PMU Event MVE_PRED.

PMU_M_EVENT_MVE_ST_MULTI_RETIRED

PMU Event MVE_ST_MULTI_RETIRED.

PMU_M_EVENT_MVE_ST_MULTI_SPEC

PMU Event MVE_ST_MULTI_SPEC.

PMU_M_EVENT_MVE_ST_RETIRED

PMU Event MVE_ST_RETIRED.

PMU_M_EVENT_MVE_ST_SPEC

PMU Event MVE_ST_SPEC.

PMU_M_EVENT_MVE_VREDUCE_FP_RETIRED

PMU Event MVE_VREDUCE_FP_RETIRED.

PMU_M_EVENT_MVE_VREDUCE_FP_SPEC

PMU Event MVE_VREDUCE_FP_SPEC.

PMU_M_EVENT_MVE_VREDUCE_INT_RETIRED

PMU Event MVE_VREDUCE_INT_RETIRED.

PMU_M_EVENT_MVE_VREDUCE_INT_SPEC

PMU Event MVE_VREDUCE_INT_SPEC.

PMU_M_EVENT_MVE_VREDUCE_RETIRED

PMU Event MVE_VREDUCE_RETIRED.

PMU_M_EVENT_MVE_VREDUCE_SPEC

PMU Event MVE_VREDUCE_SPEC.

PMU_M_EVENT_SE_CALL_NS

PMU Event SE_CALL_NS.

PMU_M_EVENT_SE_CALL_S

PMU Event SE_CALL_S.

PPB_READ_ACCESS

Trace reads from the core to memory mapped registers in the PPB (private peripheral bus) address range.
Fields:

ACCESS_SIZE `unsigned int`
: Logarithm of the access size: 0=byte, 1=halfword, 2=word, 3=doubleword, ...

BURST_LEN `unsigned int`
: The number of data transfers (beats) in this burst.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DATA unsigned int
: The data transferred.

INNER_ATTRIBUTES unsigned int
: The inner attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

INST bool
: Is an instruction fetch.

LOCK enum
: Normal / Locked / Exclusive.

MASTER_ID unsigned int
: The AXI master ID.

NS bool
: Is a nonsecure access.

OUTER_ATTRIBUTES unsigned int
: The outer attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

PADDR unsigned int
: Physical address of access.

PRIV bool
: Is privileged access.

RESPONSE enum
: Whether the transaction was successful, or an error occurred.

USER_FLAGS unsigned int
: Core specific additional signals.

PPB_WRITE_ACCESS

Trace writes from the core to memory mapped registers in the PPB (private peripheral bus) address range. Fields:

ACCESS_SIZE unsigned int
: Logarithm of the access size: 0=byte, 1=halfword, 2=word, 3=doubleword, ...

BURST_LEN unsigned int
: The number of data transfers (beats) in this burst.

DATA unsigned int
: The data transferred.

INNER_ATTRIBUTES unsigned int
: The inner attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

INST bool
: Is an instruction fetch.

LOCK enum
: Normal / Locked / Exclusive.

MASTER_ID unsigned int
: The AXI master ID.

NS bool
: Is a nonsecure access.

OUTER_ATTRIBUTES unsigned int
: The outer attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

PADDR unsigned int
: Physical address of access.

PRIV bool
: Is privileged access.

RESPONSE enum
: Whether the transaction was successful, or an error occurred.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

USER_FLAGS unsigned int
Core specific additional signals.

PRELOAD_DATA

Data preload from PLD instruction. Fields:

VADDR unsigned int
Virtual address of the data that should be preloaded.

PRELOAD_INST

Instruction preload from PLI instruction. Fields:

VADDR unsigned int
Virtual address of the instruction that should be preloaded.

PRE_CORE_LOAD

Trace just before a core load. Fields:

CURRENT_TIME unsigned int
The core's current time, as simulated time plus local time.
LOCAL_TIME unsigned int
The core's local time, relative to the current quantum.

PRE_CORE_STORE

Trace just before a core store. Fields:

CURRENT_TIME unsigned int
The core's current time, as simulated time plus local time.
LOCAL_TIME unsigned int
The core's local time, relative to the current quantum.

PSTATE_PACTIVE_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Operating mode after the transition
OLD_STATE enum
Operating mode before the transition

PSTATE_PACTIVE_OPMODE_TRANSITION_DENIED

Trace the denied pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Operating mode after the transition
OLD_STATE enum
Operating mode before the transition
REASON string
Reason for the transition being denied

PSTATE_PACTIVE_POWER_TRANSITION

Trace pactive and power mode transitions on PChannel. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

PSTATE_PACTIVE_POWER_TRANSITION_DENIED

Trace the denied pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_THREAD_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions for threads on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

RUN_STATE

Run state transition. Fields:

INST_COUNT unsigned int
: Ticks count at point of transition.

NEW enum
: New run state.

OLD enum
: Old run state.

SEMIHOSTING_CALL

Call of a semihost function occurred. Fields:

PC unsigned int
: The program counter after the semihosting call.

SEMIHOSTING_PRECALL

About to call semihost. Fields:

EL2 bool
: Use EL2 translation regime.

NS bool
: Is Non-Secure.

REG_WIDTH unsigned int
: The current register width in bytes.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SIGNAL

External signal state change. Fields:

SIGNAL enum
: Signal that changed

STATE bool
: Signal asserted state

START_COMPILE

Compilation started. Fields:

VADDR unsigned int
: Instruction where compilation begins.

STG_MULTIPLE_MICRO_OPS_ADDRESS_RANGE

Address range for MTE stores with multiple uops. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

STORE_MULTIPLE_REGS_ADDRESS_RANGE

Address range for stores with multiple registers. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

SUBOPTIMAL_LDST_RETIRED

suboptimal load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1: half-word, 2: word ...)

OPTIMAL_ALIGN_SELECTOR unsigned int
: Selector to decide the optimal alignment size (1: access width, 2: 4-bytes, 3: 8-bytes, ..., 12: 4-kbytes)

VADDR unsigned int
: The virtual address of the access

SYNC

Called for every synchronization, that is every quantum break. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

LOCAL_QUANTUM unsigned int
: The local quantum of this CPU.

LOCAL_TIME unsigned int
: The local time of this CPU.

SYNC_REASON enum
: Reason for sync.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SYSCALL

System call instruction executed. Fields:

IMM unsigned int
: Immediate value of the system call instruction.

TYPE enum
: System call type.

VADDR unsigned int
: Instruction that caused the system call.

SYSREG_READ

Trace reads from system registers. Fields:

REG_NAME enum
: Register number

VALUE unsigned int
: The value read.

SYSREG_UPDATE32

Triggers when the system updates a register. Fields:

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

SYSREG_UPDATE64

Triggers when the system updates a register. Fields:

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

SYSREG_WRITE

Trace write to system registers. Fields:

REG_ACCESS enum
: Register access status

REG_NAME enum
: Register number

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: Value written to the register.

UNALIGNED_LDST_RETIRED

Processor unaligned load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1=>half-word, 2->word ...)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NUMBER_OF_BEATS unsigned int
: Number of accesses (beats) in this burst (Total size in bytes is ACCESS_SIZE * NUMBER_OF_BEATS)

VADDR unsigned int
: The virtual address of the access

VFP_D_REGS

VFP/NEON D 64 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_Q_REGS

VFP/NEON Q 128 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding byte in the register. For example, 0x1 would indicate a write to VALUE[7:0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_SYS_REGS

Writes to the VFP/NEON units system registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: Which VFP system register is written.

MASK unsigned int
: The mask indicating updated cumulative bits.

OLD_VALUE unsigned int
: The register's old value overwritten.

VALUE unsigned int
: The new value written to the VFP system register.

VFP_S_REGS

VFP/NEON S 32 bit register write. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ALIASING enum
: Type of register aliasing used.

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

WAYPOINT

Signals end of basic block execution. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: Origin instructions set

IS_COND bool
: Indicates if this is a conditional waypoint.

PC unsigned int
: Origin address (or 0 if unavailable)

TAKEN bool
: Indicates if this waypoint was taken.

TARGET unsigned int
: Destination address

TARGET_ISET enum
: Destination instructions set

WFE_END

WFE ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFE.

WFE_EVENT_REGISTER

WFE event register status: set/clear, reason. Fields:

INST_COUNT unsigned int
: Ticks count.

REASON enum
: Reason for set/clear. Only REASON==0 (Cleared by WFE) clears the bit, all other reasons set it.

WFE_IGNORED

WFE ignored. Fields:

EVENT bool
: This WFE was ignored because the event register was set.

INST_COUNT unsigned int
: Ticks count when ignoring WFE.

TRAPPED bool
: This WFE was trapped.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

WFE_START

WFE entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFE.

WFI_END

WFI ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFI.

WFI_IGNORED

WFI ignored. Fields:

DISABLED bool
: This WFI was ignored because WFI is disabled.

INST_COUNT unsigned int
: Ticks count when ignoring WFI.

REASON enum
: specifies reason why WFI trace was ignored

TRAPPED bool
: This WFI was trapped.

WFI_START

WFI entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFI.

WFI_WAKEUP

WFI wakeup. Fields:

INST_COUNT unsigned int
: Ticks count when WFI wakeup occurred.

REASON enum
: Reason for wakeup.

XPSR

Changes to the xPSR register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

OLD_VALUE unsigned int
: The old xPSR value

UNKNOWN unsigned int
: Bits within the register that have unknown value.

VALUE unsigned int
: The new xPSR value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.37 ARM_Cortex-M7 - trace

This section describes the trace sources.

ASYNC_MEMORY_FAULT

Context ID Register write. Fields:

FAULT unsigned int
: Fault status

PADDR unsigned int
: Physical Address (or 0 if unavailable)

ATOMIC_END_ACCESS

Bus trace access for atomics. Fields:

ACCESS_FAIL bool
: Memory access failed

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

LOAD_VALUE unsigned int
: Loaded values

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ATOMIC_START_ACCESS

Bus trace access for atomics. Fields:

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ArchMsg.Error.exit_code

Fields:

EXITCODE signed int
: Exit code to be returned

KIND string
: Component kind that invoked the exit code trace

REASON string
: Reason why the exit code trace was invoked (optional)

ArchMsg.Warning.TarmacReconstructor

Unexpected event happened during Tarmac reconstruction. Fields:

REASON string
: Reason

ArchMsg.Warning.Unpredictably Indexed PM Event Register

Fields:

IsDirect unsigned int
: Direct Access rather than indirect

N unsigned int
: Number of Accessible Registers

SEL unsigned int
: Selector

ArchMsg.Warning.branch_to_unaligned_address

Fields:

ADDR unsigned int
: Unaligned branch address

ArchMsg.Warning.cache_contents_unknown

A cache has been activated without being invalidated and may contain UNKNOWN content. Fields:

REGIME unsigned int
: Current translation regime

SIDE unsigned int
: 0 d-side cache or unified cache, 1 i-side

ArchMsg.Warning.dap_csw_bad_size

A write to CM3DAP CSW has an invalid size field. Fields:

DATA unsigned int
: bits[2:0] are size

ArchMsg.Warning.dcimvac_matches_watchpoint

Watchpoints matching an AArch32 DCIMVA are implementation defined.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_fieldmismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_fracbitsnegative

VCVT Instruction is unpredictable with a negative number of fraction bits.

ArchMsg.Warning.decode_initblock

Instruction is unpredictable in an IT block.

ArchMsg.Warning.decode_initblocknotlast

Instruction is unpredictable in an IT block when not the last.

ArchMsg.Warning.decode_invalid_condition

Instruction is unpredictable when the condition is not AL.

ArchMsg.Warning.decode_invalidfieldcombination

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_invalidvalue

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_mem_hint_unallocated

There is no memory hint allocated to this bit pattern.

ArchMsg.Warning.decode_registermatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registermismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registeroutofrange

Float point instruction is unpredictable indexing beyond the end of the register bank.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_sbzsbo

Reserved bits in the instruction are not canonical.

ArchMsg.Warning.decode_transactiontoobig

Attempt to perform a memory transaction of more than 128 bytes.

ArchMsg.Warning.decode_unpred_other

Instruction is unpredictable.

ArchMsg.Warning.decode_unpreduseofpc

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13inreglist

Instruction is unpredictable with R13 in register-list.

ArchMsg.Warning.decode_useofd16orhigher

Attempt to use register D16 or higher where it is not allowed.

ArchMsg.Warning.decode_writebackandbaseinlist

Instruction is unpredictable with write back when the base register is in the transfer list.

ArchMsg.Warning.decode_zeroregistersinlist

Instruction is unpredictable when no registers are to be transfered.

ArchMsg.Warning.recursive_branch

An instruction is performing a branch that targets the same instruction. If this was intended then a WFI might be more effective. Fields:

PC unsigned int
: Address of instruction causing the branch

ArchMsg.Warning.recursive_exception

An instruction has generated an exception that targets the same instruction. Fields:

CPSR unsigned int
: Processor state

PC unsigned int
: Address of instruction causing the exception

TYPE enum
: Type of exception

ArchMsg.Warning.reentrant_vector_catch

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR unsigned int
vector address

ArchMsg.Warning.reserved_it_state

Fields:

ITSTATE unsigned int
reserved IT state

ArchMsg.Warning.secure_vector_fetch_from_nonsecure

Secure vector table is being fetched from Non-secure memory .

ArchMsg.Warning.sg_in_it_block

Arm recommends that software does not place SG (security gate instruction) inside an IT instruction block.

ArchMsg.Warning.unpredictable_a32_breakpoint

Fields:

ADDR1 unsigned int
Address
ADDR2 unsigned int
Breakpoint address
BAS unsigned int
BAS Field
IS_ADDRESS_MISMATCH_BREAKPOINT bool
Is Mismatch breakpoint, not a match breakpoint
N unsigned int
Breakpoint number
OFFSET unsigned int
Indicates which halfword actually considering into the A32 instruction

ArchMsg.Warning.unpredictable_banked_register_access

Fields:

R unsigned int
R bit
SYSM unsigned int
SYSm

ArchMsg.Warning.unpredictable_ccfail_undef

Fields:

OPCODE unsigned int
opcode of the instruction
VADDR unsigned int
Virtual address of the instruction being executed

ArchMsg.Warning.unpredictable_ici_bits

ICI bits do not apply or are architecturally unpredictable for instruction.

ArchMsg.Warning.unpredictable_t32_breakpoint

Fields:

ADDR1 unsigned int
Address

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR2 unsigned int
Breakpoint address
BAS unsigned int
BAS Field
IS_ADDRESS_MISMATCH_BREAKPOINT bool
Is Mismatch breakpoint, not a match breakpoint
N unsigned int
Breakpoint number
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_t32_vector_catch

Fields:

ADDR unsigned int
Address
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_watchpoint_far

Fields:

LOWER_BOUND unsigned int
the lowest address accessed by the instruction that triggered the watchpoint
UPPER_BOUND unsigned int
the highest watchpointed address accessed by that instruction

ArchMsg.Warning.warning_AdvSIMDExpandImmUnexpectedZero

AdvSIMDExpandImm may treat this immediate value as UNPREDICTABLE.

ArchMsg.Warning.warning_ConditionalSMC

SMC instruction has UNPREDICTABLE effects when conditional when combined with traps.

ArchMsg.Warning.warning_access_crosses_page_boundary

Access crosses page boundary.

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault_on_both_pages

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_spanning_different_memory

Fields:

ADDR unsigned int
address of access crossing page
MEMTYPE_PAGE1 signed int
First Page's memory type (0-Normal, 1-Non-Normal)
MEMTYPE_PAGE2 signed int
Second Page's memory type (0-Normal, 1-Non-Normal)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_access_wraps_around_memory

Fields:

WIDTH unsigned int
: Address width

ArchMsg.Warning.warning_bx_from_thumbee

Fields:

ADDR unsigned int
: Destination address

ArchMsg.Warning.warning_ccsidr_unimplemented_level

Fields:

CSSELR unsigned int
: current effective value of CSSELR

ArchMsg.Warning.warning_change_to_ns_when_tge_set

Attempting to change NS to 1 when HCR.TGE = 1.

ArchMsg.Warning.warning_contiguous_bit_check_abort

Fields:

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ArchMsg.Warning.warning_contiguous_bit_error

Fields:

CONTIG_ADDR unsigned int
: address of the TLB first read, with which this entry is expected to be contiguous

CONTIG_DATA unsigned int
: expected contents based on the entry at CONTIG_ADDR

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ENTRY_DATA unsigned int
: contents of conflicting TLB entry

ArchMsg.Warning.warning_cp10_cp11_mismatch

Fields:

EL unsigned int
: EL owning the control

ArchMsg.Warning.warning_cp10_cp11_reserved_value

Fields:

CP10 unsigned int
: CP10 access

CP11 unsigned int
: CP11 access

ArchMsg.Warning.warning_csselr_level_out_of_range

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

IMPLEMENTED unsigned int
: number of cache levels implemented

LEVEL unsigned int
: written value of CSSELR.Level

TYPE enum
: The type of cache selected by CSSELR.InD and TnD

ArchMsg.Warning.warning_debug_flow_control_bits_not_obeyed

Fields:

ERROR enum
: type of error resulting

REGISTER enum
: register under consideration

ArchMsg.Warning.warning_debug_register_access_during_reset

Fields:

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

ArchMsg.Warning.warning_decode_cps_inconsistent_fields

The mode setting fields in the CPS instruction are inconsistent.

ArchMsg.Warning.warning_decode_invalid_state

Instruction is unpredictable in current exception-level/security state.

ArchMsg.Warning.warning_default_cacheable_mmu_on

Default cacheable is enabled (HCR.DC = 1) while MMU is enabled.

ArchMsg.Warning.warning_default_cacheable_vmmu_off

Default cacheable is enabled (HCR.DC = 1) while HCR.VM = 0.

ArchMsg.Warning.warning_deprecated_wvr_bit_2_set

Fields:

N unsigned int
: index

WVR unsigned int
: DBGWVR

ArchMsg.Warning.warning_eret_while_software_step_active_pending

Missing ISB between setting MDSCR_EL1.SS and ERET.

ArchMsg.Warning.warning_exclusive_to_non_normal

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_exclusive_to_non_writeback

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR unsigned int
Address

ArchMsg.Warning.warning_execute_from_device_memory

An attempt was made to execute from Device memory.

ArchMsg.Warning.warning_illegal_cpsr_mode

Fields:

NEW_MODE unsigned int
New value of CPSR.M

ArchMsg.Warning.warning_illegal_srs_mode

Fields:

MODE unsigned int
mode for Banked SP

ArchMsg.Warning.warning_implementation_defined_read_debug_register_in_SCS

Read Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

PPB_OFFSET unsigned int
debug register offset

ArchMsg.Warning.warning_implementation_defined_sequential_security_transitions_supported

The behaviour of a sequential instruction fetches that cross from non-secure to secure memory and contain SG instruction is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_implementation_defined_stack_limit_check_supported

It is IMPLEMENTATION DEFINED whether stack pointer limit checking is performed for the this instructions. Fields:

ADDRESS unsigned int
Address of instruction
OPCODE unsigned int
instruction opcode

ArchMsg.Warning.warning_implementation_defined_write_debug_register_in_SCS

Write Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

DATA unsigned int
data attempted to be written
PPB_OFFSET unsigned int
debug register offset

ArchMsg.Warning.warning_invalid_tcr_granule

Fields:

REQUEST signed int
page size requested (or 0 for reserved)
SUBSTITUTE signed int
best guess available page size
TG_ID bool
bits TG0 or TG1

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_load_multiple_user_registers_from_user_mode

An LDM(user registers) instruction executed from user mode.

ArchMsg.Warning.warning_load_pc_from_unaligned

PC loaded from an unaligned location.

ArchMsg.Warning.warning_mte_in_mem_attr_not_supported

Fields:

ATTR unsigned int
: Memory attribute

ArchMsg.Warning.warning_reserved_breakpoint_state_match

Fields:

IS_BREAKPOINT unsigned int
: Is breakpoint, not watchpoint

ArchMsg.Warning.warning_shareability

Unpredictable: combination of the 1st and 2nd stages of translation is Normal Inner Non-Cacheable, Outer Non-Cacheable.

ArchMsg.Warning.warning_software_step_set_while_enabled

MDSCR_EL1.SS set to 1 while software step debug exceptions are enabled.

ArchMsg.Warning.warning_thumb_instruction_wraps_around_memory

Thumb instruction warps around memory.

ArchMsg.Warning.warning_ttbr_sbz_bits_are_not_zero

Fields:

MSB unsigned int
: x-1

TTBR enum
: which TTBR register

ArchMsg.Warning.warning_unaligned_address_dbgdtrrx_write

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_unaligned_address_dbgdtrtx_read

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_unaligned_to_device

Fields:

ADDR unsigned int
: Address

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unaligned_to_strongly_ordered

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_unknown_sau_rnr

SAU_RNR was set to an unsupported value.

ArchMsg.Warning.warning_unpredictable_AIRCR_PRIP_and_BFHFNMINP

The effect of setting both AIRCR.BFHFNMINP and AIRCR.PRIP to 1 is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTCLRACTIVE_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTCLRACTIVE if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_and_SYSRESETREQ

When the processor is halted in Debug state, if a write to the register writes a 1 to both VECTRESET and SYSRESETREQ, the behavior is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTRESET if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_incorrect_VKEY

The value 0x05FA must be written to AIRCR.VKEY, otherwise the register write is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_EXE_RETURN_Reserved_Bit

EXC_RETURN[23:7] are reserved with the special condition that all bits should be written as one. Values other than all 1s are UNPREDICTABLE. Fields:

EXC_RETURN unsigned int
: Special PC value

SBO_MASK unsigned int
: Which bits Should Be One to not UNPRED

ArchMsg.Warning.warning_unpredictable_change_to_DEMCR_MON_STEP_at_insufficient_priority

The effect of changing DEMCR.MON_STEP at an execution priority that is lower than the priority of the DebugMonitor exception is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_change_to_priority_of_active_exception

Changing the priority of an active exception is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_clear_lspact

Fields:

DRVEXC_TAKEN unsigned int
: derived exception taken not pended

REACHED_LAST_STORE unsigned int
: only the last faulted

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_dbg_exit_unaligned_dlr

Debug exit to AArch32 with DLR[0] set to 1. PC[0] can be set to 0 or to DLR[0].

ArchMsg.Warning.warning_unpredictable_exception_catch

Generation of an exception catch event is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_inconsistent_state

state on exception return is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_instruction

exception return instruction is unpredictable.

ArchMsg.Warning.warning_unpredictable_in_debug_state

Instruction is UNPREDICTABLE when executed in debug state.

ArchMsg.Warning.warning_unpredictable_mair_encoding

Fields:

ATTR `unsigned int`
: Memory attribute

ArchMsg.Warning.warning_unpredictable_pmu_counter_access

Access to PMU counters from non-secure EL0 or EL1 is UNPREDICTABLE with MDCR_EL2.HPMN set to 0.

ArchMsg.Warning.warning_unpredictable_prioritization_breakpoint_match_vector_catch

Two events with the same priority occurred at the same instruction: Address Matching Vector Catch debug event, and Breakpoint debug event It is constrained unpredictable which is taken.

ArchMsg.Warning.warning_unpredictable_stack_selection

UNPREDICTABLE state on context switch.

ArchMsg.Warning.warning_unpredictable_tsize_out_of_range

Fields:

TCR `enum`
: Which TCR

TSIZE `signed int`
: Value of TSize

TSIZE_MAX `signed int`
: The maximum allowed value

TTBR `unsigned int`
: Which TTBR region

ArchMsg.Warning.warning_unpredictable_unaligned_pc_as_base_register

The PC value must be word-aligned, otherwise the behavior of the instruction is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_unaligned_pop_stack

Fields:

FRAME_ADDR `unsigned int`
: unaligned stack frame address

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_vmsa_memattrib

Fields:

TEXCB unsigned int
: texcb value

TEX_REMAP bool
: use tex remap

ArchMsg.Warning.warning_unpredictable_vtcr_t0sz_sl0

Fields:

SL0 signed int
: VTCR.SL0, starting level for stage 2 translation table walks

T0SZ signed int
: VTCR.T0SZ, required input address range

ArchMsg.Warning.warning_unpredictable_write_DHCSR

Fields:

NEW unsigned int
: data attempted to be written

OLD unsigned int
: previous value

ArchMsg.Warning.warning_unpredictable_write_SHCSR

UNPREDICTABLE setting a pending bit to 1 for an exception with priority >= execution priority.

ArchMsg.Warning.warning_unsupported_access_to_memory_mapped_register

Access to memory mapped register is unsupported.

ArchMsg.Warning.warning_user_jmcr_access

Fields:

WRITE unsigned int
: access is write

ArchMsg.Warning.warning_wcr_mask_and_bas

DBGWCR_EL1.MASK is non-zero and BAS is not 0xff.

ArchMsg.Warning.warning_wcr_mask_reserved

DBGWCR_ELn.MASK is set to a reserved value.

ArchMsg.Warning.warning_wcr_non_configuous_bas

DBGWCR_EL1.BAS has non-contiguous set of ones.

ArchMsg.Warning.warning_wvr_masked_bit_not_zero

DBGWVRn_EL1.VA contains a bit masked by DBGWCRn_EL1.MASK that is not zero.

ArchMsg.warning_unpred_system_register_access

Fields:

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

BRANCH_MISPREDICT

Simulating branch mispredict. Fields:

PC unsigned int
: Origin address (or 0 if unavailable)

BRA_DIR

Direct branches, to immediate address. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR

Indirect branches, perhaps to a register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR_CCFAIL

Indirect branches fail, perhaps to a register.

BRA_RET_CCFAIL

Branche return fail, perhaps to a register.

CACHE_MAINTENANCE_OP

Cache Maintenance Operation. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Specified MVA or Set/way

FUNCTION enum
: Clean / Invalidate

SCOPE enum
: Affected region

SIDE enum
: Inst / Data.

CCFAIL

Conditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCFAIL_UNC

Unconditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS

Conditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS_UNC

Unconditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CODE_CACHE_MAINT

Code cache maintenance. Fields:

END_INCL unsigned int
: Inclusive end addres for by VA or PA requests

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS bool
: Non-secure world for PA

START unsigned int
: Start address for by VA or PA requests

TYPE enum
: Request type.

COMPILE_BLOCK_END

Last instruction of basic block translated. Fields:

VADDR unsigned int
: Address of next instruction after this basic block

COMPILE_BLOCK_START

First instruction of basic block translated. Fields:

VADDR unsigned int
: Address of first instruction in basic block

COMPILE_INST

ARM instruction compiled. Fields:

DISASS string
: Disassembly of the instruction.

ISET enum
: The instruction set of this instruction.

ITSTATE unsigned int
: The ITSTATE current for the instruction.

OPCODE unsigned int
: The opcode of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

CONTEXTIDR

Context ID Register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

NS bool
: Secure or nonsecure banked register is accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

CONTEXT_SYNC

Called for every context synchronization event. Fields:

ARCH_CSE bool
: Architectural Context Synchronization Event

SYSREG_SYNC bool
: System register synchronization event

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_ENDIAN

Core BE8 Big-Endian state changed. Fields:

BE8 bool
: Core BE8 Big-Endian state

CORE_INFO

Static processor attributes. Only triggered by a call to DumpState(). Fields:

ARCH_PROFILE enum
: The architecture profile of the core.

CLUSTER_ID unsigned int
: The cluster ID of this processor.

CORE_NUM unsigned int
: The number of this core in an MP processor.

DETAILED_CORE_FEATURES enum
: Detailed core features as a bitfield. Enum values are bitmasks to extract the info from the field

FPU_VERSION enum
: The VFP version implemented by the core.

MEM_ARCH enum
: The memory architecture of the core.

MIDR_EL1 unsigned int
: MIDR_EL1 value

NUM_CORES unsigned int
: The number of cores in this MP processor.

QUANTUM_SIZE unsigned int
: The default quantum size of the core.

SECURITY_FEATURES bool
: Does the core have security features?

CORE_LOADS

Processor load accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CORE_REGS

Changes of the core registers R0 to R14. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number, 0 to 14.

OLD_VALUE unsigned int
: The old value overwritten.

PHYS_ID enum
: The physical register accessed.

VALUE unsigned int
: The new value written to the register.

CORE_STORES

Processor store accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CRYPTO_SPEC

Every crypto instruction speculatively executed.

DATA_CACHE_SET_ALLOC_TAGS

Set allocation tags from DC GVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of memory in bytes.

TAG unsigned int
: The value of the tag being set.

VADDR unsigned int
: Virtual address used for setting allocation tags.

DATA_CACHE_ZERO

Zero and invalidate cache line from DC ZVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of zero'd memory in bytes.

VADDR unsigned int
: Virtual address of the zero'd cache line.

DEBUG_EVENT

Hardware debug support event. Fields:

EVENT enum
: Description of event

VALUE unsigned int
: data value

DTCM_ACCESS

Trace TCM accesses.

DTCM_ACCESS_RAZ_WI

Trace TCM accesses. Fields:

ADDR unsigned int
: Address of the RAZ/WI access.

DWT_MATCH

DWT comparator matches. Fields:

NUM unsigned int
: DWT comparator number

TYPE enum
: DWT comparator configuration

END_COMPILE

Compilation end. Fields:

END_OF_PAGE_COUNT unsigned int
: Number of basic blocks exited due to page boundary since START_COMPILE.

FETCHFAIL_COUNT unsigned int
: Number of basic blocks exited due to ifetch failure START_COMPILE.

INST_COUNT unsigned int
: Number of instructions compiled since START_COMPILE.

NONSEQ_COUNT unsigned int
: Number of basic blocks exited due to non-sequential instructions since START_COMPILE.

PAGE_STRADDLE_COUNT unsigned int
: Number of basic blocks exited due to unaligned instructions crossing page since START_COMPILE.

EXCEPTION

Exceptions which are taken. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: The instruction set of the processor when the exception occurred.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LR unsigned int
: The value assigned to the link register.

PC unsigned int
: The location where the exception occurred.

TARGET_ISET enum
: The instruction set of the exception handler code.

TARGET_PC unsigned int
: The address the exception branches to.

VECTOR enum
: The exception vector.

EXCEPTION_END

Every exception completed.

EXCEPTION_ENTRY

Event marking the entry of an exception. All pushes of register to the stack will follow this event. Fields:

PC unsigned int
: The location where the exception occurred.

VECTOR enum
: The exception that occurred.

EXCEPTION_RAISE

Every exception raised.

EXCEPTION_RETURN

Marks the end of an exception, but on an M core does not indicate a program flow change (branch). In most situations it will be followed by an indirect branch. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

PC unsigned int
: The address of the branch instruction.

EXCEPTION_RETURN_PREBRANCH

Exception return event that must occur before PC and processor state have updated.

EXCEPTION_START

Every exception started.

EXCEPTION_VECTOR_FETCH

I-side vector fetch for M-class exception. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Result of fetch if OK

RESPONSE enum
: 0=Aborted, 1=OK

VADDR unsigned int
: Where the vector is fetched from (including VTOR)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

FP_STATE

Floating point state. Fields:

FPSR unsigned int
: FPSR register value.

MASK unsigned int
: The mask indicating updated cumulative bits.

HLT

HLT instruction occurred. Fields:

IMM16 unsigned int
: The 16-bit immediate value encoded in HLT instruction.

INFO_EXCEPTION_REASON

Provide information of real cause of exception. Fields:

FaultCause enum
: Which bit of which V7M/V8M-mainline FSR/HFSR would have been set.

PC unsigned int
: The location where the exception occurred.

PHASE enum
: What we are doing with the exception.

REASONS enum
: Why we are doing it.

VECTOR enum
: The exception that occurred.

INFO_FP_CONTEXT

Inform when ExecuteFPCheck writes CONTROL.FPCA and FPSCR etc. Fields:

WHAT enum
: What we are doing.

INFO_STACKING

Provide information of real cause of exception. Fields:

PHASE enum
: Are we starting or ending a sequence of stack writes.

REASONS enum
: Why we are doing it.

INST

Every instruction executed. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DEBUG_STATE bool
: The instruction is executed in debug state.

DISASS string
: Disassembly of instruction.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The current instruction set.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ITSTATE unsigned int
: The current ITSTATE.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

MODE enum
: The mode the core is in.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

OPCODE unsigned int
: The opcode of the instruction.

PADDR unsigned int
: The physical address of the instruction.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

INST_END

Every instruction completed.

INST_START

Every instruction started. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

INST_STRADDLE

Instruction straddles boundary. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

ISB

ISB instruction occurred.

ITCM_ACCESS

Trace TCM accesses.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ITCM_ACCESS_RAZ_WI

Trace TCM accesses. Fields:

ADDR unsigned int
: Address of the RAZ/WI access.

ITM

Instrumentation Trace Macrocell. Fields:

ITM_PACKET_TYPE enum
: ITM and DWT packets type.

PACKET_HEADER unsigned int
: ITM Packet Header.

PACKET_PAYLOAD unsigned int
: ITM Packet Payload.

LOAD_MULTIPLE_REGS_LIST

List of destination registers (not source) of a multiple registers load instruction. Fields:

REG string
: Register to add into the list

LOCAL_MONITOR

Local monitor activity. Fields:

PADDR unsigned int
: Local Monitor Address

State enum
: State of the monitor (Open/Exclusive)

LOCKUP_CYCLE

This event is triggered if the core enters or stays in the lock-up state. Fields:

CAUSE enum
: What causes the lockup?.

LOCKUP_ENTRY

This event is triggered if the core enters the lock-up state. Fields:

PC unsigned int
: The PC when the lockup state is entered.

LOCKUP_EXIT

This event is triggered if the core leaves the lock-up state.

MODE_CHANGE

Mode change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: The new mode.

OLD_MODE enum
: The old mode.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MPU_TRANS

Address translation information. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

EXEC_PERM enum
: Execution Permission

INNERCACHE_RA bool
: Is the inner cache allocate on read

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

MEMTYPE enum
: Memory type.

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

PADDR unsigned int
: Address of the access.

PAGESIZE unsigned int
: Page size as log2(size).

READ_PERM enum
: Read Permission

REG_NUM signed int
: MPU region-number 0-15 (else 0xFF=none, 0xBB/0xDD=background/default)

SH enum
: Shareability

SIDE enum
: Inst / Data.

WRITE_PERM enum
: Write Permission

PERIODIC

Called for every quantum. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

PC unsigned int
: The address of the next instruction to be executed on this CPU.

PPB_READ_ACCESS

Trace reads from the core to memory mapped registers in the PPB (private peripheral bus) address range. Fields:

ACCESS_SIZE unsigned int
: Logarithm of the access size: 0=byte, 1=halfword, 2=word, 3=doubleword, ...

BURST_LEN unsigned int
: The number of data transfers (beats) in this burst.

DATA unsigned int
: The data transferred.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INNER_ATTRIBUTES unsigned int
: The inner attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

INST bool
: Is an instruction fetch.

LOCK enum
: Normal / Locked / Exclusive.

MASTER_ID unsigned int
: The AXI master ID.

NS bool
: Is a nonsecure access.

OUTER_ATTRIBUTES unsigned int
: The outer attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

PADDR unsigned int
: Physical address of access.

PRIV bool
: Is privileged access.

RESPONSE enum
: Whether the transaction was successful, or an error occurred.

USER_FLAGS unsigned int
: Core specific additional signals.

PPB_WRITE_ACCESS

Trace writes from the core to memory mapped registers in the PPB (private peripheral bus) address range. Fields:

ACCESS_SIZE unsigned int
: Logarithm of the access size: 0=byte, 1=halfword, 2=word, 3=doubleword, ...

BURST_LEN unsigned int
: The number of data transfers (beats) in this burst.

DATA unsigned int
: The data transferred.

INNER_ATTRIBUTES unsigned int
: The inner attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

INST bool
: Is an instruction fetch.

LOCK enum
: Normal / Locked / Exclusive.

MASTER_ID unsigned int
: The AXI master ID.

NS bool
: Is a nonsecure access.

OUTER_ATTRIBUTES unsigned int
: The outer attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

PADDR unsigned int
: Physical address of access.

PRIV bool
: Is privileged access.

RESPONSE enum
: Whether the transaction was successful, or an error occurred.

USER_FLAGS unsigned int
: Core specific additional signals.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PRELOAD_DATA

Data preload from PLD instruction. Fields:

VADDR unsigned int
: Virtual address of the data that should be preloaded.

PRELOAD_INST

Instruction preload from PLI instruction. Fields:

VADDR unsigned int
: Virtual address of the instruction that should be preloaded.

PRE_CORE_LOAD

Trace just before a core load. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_CORE_STORE

Trace just before a core store. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PSTATE_PACTIVE_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

PSTATE_PACTIVE_OPMODE_TRANSITION_DENIED

Trace the denied pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_POWER_TRANSITION

Trace pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OLD_STATE enum
: Power mode before the transition

PSTATE_PACTIVE_POWER_TRANSITION_DENIED

Trace the denied pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_THREAD_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions for threads on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

RUN_STATE

Run state transition. Fields:

INST_COUNT unsigned int
: Ticks count at point of transition.

NEW enum
: New run state.

OLD enum
: Old run state.

SEMIHOSTING_CALL

Call of a semihost function occurred. Fields:

PC unsigned int
: The program counter after the semihosting call.

SEMIHOSTING_PRECALL

About to call semihost. Fields:

EL2 bool
: Use EL2 translation regime.

NS bool
: Is Non-Secure.

REG_WIDTH unsigned int
: The current register width in bytes.

SIGNAL

External signal state change. Fields:

SIGNAL enum
: Signal that changed

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

STATE bool
: Signal asserted state

START_COMPILE

Compilation started. Fields:

VADDR unsigned int
: Instruction where compilation begins.

STG_MULTIPLE_MICRO_OPS_ADDRESS_RANGE

Address range for MTE stores with multiple uops. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

STORE_MULTIPLE_REGS_ADDRESS_RANGE

Address range for stores with multiple registers. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

SUBOPTIMAL_LDST_RETIRED

suboptimal load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1: half-word, 2: word ...)

OPTIMAL_ALIGN_SELECTOR unsigned int
: Selector to decide the optimal alignment size (1: access width, 2: 4-bytes, 3: 8-bytes, ..., 12: 4-kbytes)

VADDR unsigned int
: The virtual address of the access

SYNC

Called for every synchronization, that is every quantum break. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

LOCAL_QUANTUM unsigned int
: The local quantum of this CPU.

LOCAL_TIME unsigned int
: The local time of this CPU.

SYNC_REASON enum
: Reason for sync.

SYSCALL

System call instruction executed. Fields:

IMM unsigned int
: Immediate value of the system call instruction.

TYPE enum
: System call type.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VADDR unsigned int
: Instruction that caused the system call.

SYSREG_READ

Trace reads from system registers. Fields:

REG_NAME enum
: Register number

VALUE unsigned int
: The value read.

SYSREG_UPDATE32

Triggers when the system updates a register. Fields:

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

SYSREG_UPDATE64

Triggers when the system updates a register. Fields:

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

SYSREG_WRITE

Trace write to system registers. Fields:

REG_ACCESS enum
: Register access status

REG_NAME enum
: Register number

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: Value written to the register.

UNALIGNED_LDST_RETIRED

Processor unaligned load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1=>half-word, 2->word ...)

NUMBER_OF_BEATS unsigned int
: Number of accesses (beats) in this burst (Total size in bytes is ACCESS_SIZE * NUMBER_OF_BEATS)

VADDR unsigned int
: The virtual address of the access

VFP_D_REGS

VFP/NEON D 64 bit register write. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_Q_REGS

VFP/NEON Q 128 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding byte in the register. For example, 0x1 would indicate a write to VALUE[7:0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_SYS_REGS

Writes to the VFP/NEON units system registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: Which VFP system register is written.

MASK unsigned int
: The mask indicating updated cumulative bits.

OLD_VALUE unsigned int
: The register's old value overwritten.

VALUE unsigned int
: The new value written to the VFP system register.

VFP_S_REGS

VFP/NEON S 32 bit register write. Fields:

ALIASING enum
: Type of register aliasing used.

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

WAYPOINT

Signals end of basic block execution. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: Origin instructions set

IS_COND bool
: Indicates if this is a conditional waypoint.

PC unsigned int
: Origin address (or 0 if unavailable)

TAKEN bool
: Indicates if this waypoint was taken.

TARGET unsigned int
: Destination address

TARGET_ISET enum
: Destination instructions set

WFE_END

WFE ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFE.

WFE_EVENT_REGISTER

WFE event register status: set/clear, reason. Fields:

INST_COUNT unsigned int
: Ticks count.

REASON enum
: Reason for set/clear. Only REASON==0 (Cleared by WFE) clears the bit, all other reasons set it.

WFE_IGNORED

WFE ignored. Fields:

EVENT bool
: This WFE was ignored because the event register was set.

INST_COUNT unsigned int
: Ticks count when ignoring WFE.

TRAPPED bool
: This WFE was trapped.

WFE_START

WFE entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

WFI_END

WFI ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFI.

WFI_IGNORED

WFI ignored. Fields:

DISABLED bool
: This WFI was ignored because WFI is disabled.

INST_COUNT unsigned int
: Ticks count when ignoring WFI.

REASON enum
: specifies reason why WFI trace was ignored

TRAPPED bool
: This WFI was trapped.

WFI_START

WFI entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFI.

WFI_WAKEUP

WFI wakeup. Fields:

INST_COUNT unsigned int
: Ticks count when WFI wakeup occurred.

REASON enum
: Reason for wakeup.

XPSR

Changes to the xPSR register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

OLD_VALUE unsigned int
: The old xPSR value

UNKNOWN unsigned int
: Bits within the register that have unknown value.

VALUE unsigned int
: The new xPSR value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.38 ARM_Cortex-R4 - trace

This section describes the trace sources.

ASYNC_MEMORY_FAULT

Context ID Register write. Fields:

FAULT unsigned int
: Fault status

PADDR unsigned int
: Physical Address (or 0 if unavailable)

ATOMIC_END_ACCESS

Bus trace access for atomics. Fields:

ACCESS_FAIL bool
: Memory access failed

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

LOAD_VALUE unsigned int
: Loaded values

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ATOMIC_START_ACCESS

Bus trace access for atomics. Fields:

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ArchMsg.Error.exit_code

Fields:

EXITCODE signed int
: Exit code to be returned

KIND string
: Component kind that invoked the exit code trace

REASON string
: Reason why the exit code trace was invoked (optional)

ArchMsg.Warning.TarmacReconstructor

Unexpected event happened during Tarmac reconstruction. Fields:

REASON string
: Reason

ArchMsg.Warning.cache_contents_unknown

A cache has been activated without being invalidated and may contain UNKNOWN content. Fields:

REGIME unsigned int
: Current translation regime

SIDE unsigned int
: 0 d-side cache or unified cache, 1 i-side

ArchMsg.Warning.recursive_branch

An instruction is performing a branch that targets the same instruction. If this was intended then a WFI might be more effective. Fields:

PC unsigned int
: Address of instruction causing the branch

ArchMsg.Warning.recursive_exception

An instruction has generated an exception that targets the same instruction. Fields:

CPSR unsigned int
: Processor state

PC unsigned int
: Address of instruction causing the exception

TYPE enum
: Type of exception

ArchMsg.Warning.secure_vector_fetch_from_nonsecure

Secure vector table is being fetched from Non-secure memory .

BRANCH_MISPREDICT

Simulating branch mispredict. Fields:

PC unsigned int
: Origin address (or 0 if unavailable)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

BRA_DIR

Direct branches, to immediate address. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR

Indirect branches, perhaps to a register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR_CCFAIL

Indirect branches fail, perhaps to a register.

BRA_RET_CCFAIL

Branche return fail, perhaps to a register.

CACHE_MAINTENANCE_OP

Cache Maintenance Operation. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Specified MVA or Set/way

FUNCTION enum
: Clean / Invalidate

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SCOPE enum
: Affected region

SIDE enum
: Inst / Data.

CCFAIL

Conditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCFAIL_UNC

Unconditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS

Conditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS_UNC

Unconditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CODE_CACHE_MAINT

Code cache maintenance. Fields:

END_INCL unsigned int
: Inclusive end addres for by VA or PA requests

NS bool
: Non-secure world for PA

START unsigned int
: Start address for by VA or PA requests

TYPE enum
: Request type.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

COMPILE_BLOCK_END

Last instruction of basic block translated. Fields:

VADDR unsigned int
: Address of next instruction after this basic block

COMPILE_BLOCK_START

First instruction of basic block translated. Fields:

VADDR unsigned int
: Address of first instruction in basic block

COMPILE_INST

ARM instruction compiled. Fields:

DISASS string
: Disassembly of the instruction.

ISET enum
: The instruction set of this instruction.

ITSTATE unsigned int
: The ITSTATE current for the instruction.

OPCODE unsigned int
: The opcode of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

CONTEXTIDR

Context ID Register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

NS bool
: Secure or nonsecure banked register is accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

CONTEXT_SYNC

Called for every context synchronization event. Fields:

ARCH_CSE bool
: Architectural Context Synchronization Event

SYSREG_SYNC bool
: System register synchronization event

CORE_ENDIAN

Core BE8 Big-Endian state changed. Fields:

BE8 bool
: Core BE8 Big-Endian state

CORE_INFO

Static processor attributes. Only triggered by a call to DumpState(). Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ARCH_PROFILE enum
: The architecture profile of the core.

CLUSTER_ID unsigned int
: The cluster ID of this processor.

CORE_NUM unsigned int
: The number of this core in an MP processor.

DETAILED_CORE_FEATURES enum
: Detailed core features as a bitfield. Enum values are bitmasks to extract the info from the field

FPU_VERSION enum
: The VFP version implemented by the core.

MEM_ARCH enum
: The memory architecture of the core.

MIDR_EL1 unsigned int
: MIDR_EL1 value

NUM_CORES unsigned int
: The number of cores in this MP processor.

QUANTUM_SIZE unsigned int
: The default quantum size of the core.

SECURITY_FEATURES bool
: Does the core have security features?

CORE_LOADS

Processor load accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CORE_REGS

Changes of the core registers R0 to R14. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number, 0 to 14.

MODE enum
: Bank of the register accessed.

OLD_VALUE unsigned int
: The old value overwritten.

PHYS_ID enum
: The physical register accessed.

VALUE unsigned int
: The new value written to the register.

CORE_STORES

Processor store accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CP14_READ

Reads to the System Coprocessor registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP14_WRITE

Writes to the System Coprocessor registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CP15_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

CP15_WRITE

System Control Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

CPSR

CPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

OLD_VALUE unsigned int
: The old CPSR value

UNKNOWN unsigned int
: Bits within the register that have unknown value.

VALUE unsigned int
: The new CPSR value

CRYPTO_SPEC

Every crypto instruction speculatively executed.

DATA_CACHE_SET_ALLOC_TAGS

Set allocation tags from DC GVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of memory in bytes.

TAG unsigned int
: The value of the tag being set.

VADDR unsigned int
: Virtual address used for setting allocation tags.

DATA_CACHE_ZERO

Zero and invalidate cache line from DC ZVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of zero'd memory in bytes.

VADDR unsigned int
: Virtual address of the zero'd cache line.

DTCM_ACCESS

Trace TCM accesses.

DTCM_ACCESS_RAZ_WI

Trace TCM accesses. Fields:

ADDR unsigned int
: Address of the RAZ/WI access.

END_COMPILE

Compilation end. Fields:

END_OF_PAGE_COUNT unsigned int
: Number of basic blocks exited due to page boundary since START_COMPILE.

FETCHFAIL_COUNT unsigned int
: Number of basic blocks exited due to ifetch failure START_COMPILE.

INST_COUNT unsigned int
: Number of instructions compiled since START_COMPILE.

NONSEQ_COUNT unsigned int
: Number of basic blocks exited due to non-sequential instructions since START_COMPILE.

PAGE_STRADDLE_COUNT unsigned int
: Number of basic blocks exited due to unaligned instructions crossing page since START_COMPILE.

EXCEPTION

Exception taken. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

IS_PHYSICAL bool
: Physical or Virtual exception

LR unsigned int
: The value assigned to the link register.

PC unsigned int
: The location where the exception occurred.

PREFERRED_RETURN unsigned int
: The preferred return address for the exception.

TARGET_ISET enum
: The instruction set of the exception handler code.

TARGET_PC unsigned int
: The address the exception branches to.

VECTOR enum
: The exception vector.

VECTOR_OFFSET unsigned int
: The offset of an exception vector from the base address.

EXCEPTION_END

Every exception completed.

EXCEPTION_RAISE

Every exception raised.

EXCEPTION_RETURN

Branches on leaving exception. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

EXCEPTION_RETURN_PREBRANCH

Exception return event that must occur before PC and processor state have updated.

EXCEPTION_START

Every exception started.

FP_STATE

Floating point state. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

FPSR unsigned int
: FPSR register value.

MASK unsigned int
: The mask indicating updated cumulative bits.

HLT

HLT instruction occurred. Fields:

IMM16 unsigned int
: The 16-bit immediate value encoded in HLT instruction.

INST

Every instruction executed. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DEBUG_STATE bool
: The instruction is executed in debug state.

DISASS string
: Disassembly of instruction.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The current instruction set.

ITSTATE unsigned int
: The current ITSTATE.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

MODE enum
: The mode the core is in.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

OPCODE unsigned int
: The opcode of the instruction.

PADDR unsigned int
: The physical address of the instruction.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

INST_END

Every instruction completed.

INST_START

Every instruction started. Fields:

ISET enum
: The current instruction set.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

INST_STRADDLE

Instruction straddles boundary. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

ISB

ISB instruction occurred.

ITCM_ACCESS

Trace TCM accesses.

ITCM_ACCESS_RAZ_WI

Trace TCM accesses. Fields:

ADDR unsigned int
: Address of the RAZ/WI access.

LOAD_MULTIPLE_REGS_LIST

List of destination registers (not source) of a multiple registers load instruction. Fields:

REG string
: Register to add into the list

LOCAL_MONITOR

Local monitor activity. Fields:

PADDR unsigned int
: Local Monitor Address

State enum
: State of the monitor (Open/Exclusive)

MODE_CHANGE

Mode change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: The new mode.

OLD_MODE enum
: The old mode.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MPU_TRANS

Address translation information. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

EXEC_PERM enum
: Execution Permission

INNERCACHE_RA bool
: Is the inner cache allocate on read

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

MEMTYPE enum
: Memory type.

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

PADDR unsigned int
: Address of the access.

PAGESIZE unsigned int
: Page size as log2(size).

READ_PERM enum
: Read Permission

REG_NUM signed int
: MPU region-number 0-15 (else 0xFF=none, 0xBB/0xDD=background/default)

SH enum
: Shareability

SIDE enum
: Inst / Data.

WRITE_PERM enum
: Write Permission

PERIODIC

Called for every quantum. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

PC unsigned int
: The address of the next instruction to be executed on this CPU.

PRELOAD_DATA

Data preload from PLD instruction. Fields:

VADDR unsigned int
: Virtual address of the data that should be preloaded.

PRELOAD_INST

Instruction preload from PLI instruction. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VADDR unsigned int
: Virtual address of the instruction that should be preloaded.

PRE_CORE_LOAD

Trace just before a core load. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_CORE_STORE

Trace just before a core store. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PSTATE_PACTIVE_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

PSTATE_PACTIVE_OPMODE_TRANSITION_DENIED

Trace the denied pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_POWER_TRANSITION

Trace pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

PSTATE_PACTIVE_POWER_TRANSITION_DENIED

Trace the denied pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_THREAD_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions for threads on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

RUN_STATE

Run state transition. Fields:

INST_COUNT unsigned int
: Ticks count at point of transition.

NEW enum
: New run state.

OLD enum
: Old run state.

SEMIHOSTING_CALL

Call of a semihost function occurred. Fields:

PC unsigned int
: The program counter after the semihosting call.

SEMIHOSTING_PRECALL

About to call semihost. Fields:

EL2 bool
: Use EL2 translation regime.

NS bool
: Is Non-Secure.

REG_WIDTH unsigned int
: The current register width in bytes.

SIGNAL

External signal state change. Fields:

SIGNAL enum
: Signal that changed

STATE bool
: Signal asserted state

SPSR

SPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MODE enum
: Which of the banked SPSR registers is written.

OLD_VALUE unsigned int
: The old SPSR value

VALUE unsigned int
: The new SPSR value

START_COMPILE

Compilation started. Fields:

VADDR unsigned int
: Instruction where compilation begins.

STG_MULTIPLE_MICRO_OPS_ADDRESS_RANGE

Address range for MTE stores with multiple uops. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

STORE_MULTIPLE_REGS_ADDRESS_RANGE

Address range for stores with multiple registers. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

SUBOPTIMAL_LDST_RETIRED

suboptimal load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1: half-word, 2: word ...)

OPTIMAL_ALIGN_SELECTOR unsigned int
: Selector to decide the optimal alignment size (1: access width, 2: 4-bytes, 3: 8-bytes, ..., 12: 4-kbytes)

VADDR unsigned int
: The virtual address of the access

SYNC

Called for every synchronization, that is every quantum break. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

LOCAL_QUANTUM unsigned int
: The local quantum of this CPU.

LOCAL_TIME unsigned int
: The local time of this CPU.

SYNC_REASON enum
: Reason for sync.

SYSCALL

System call instruction executed. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

IMM unsigned int
: Immediate value of the system call instruction.

TYPE enum
: System call type.

VADDR unsigned int
: Instruction that caused the system call.

SYSREG_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE32

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SYSREG_UPDATE64

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

UNALIGNED_LDST_RETIRED

Processor unaligned load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1=>half-word, 2->word ...)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NUMBER_OF_BEATS unsigned int
: Number of accesses (beats) in this burst (Total size in bytes is ACCESS_SIZE * NUMBER_OF_BEATS)

VADDR unsigned int
: The virtual address of the access

VFP_D_REGS

VFP/NEON D 64 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_Q_REGS

VFP/NEON Q 128 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding byte in the register. For example, 0x1 would indicate a write to VALUE[7:0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_SYS_REGS

Writes to the VFP/NEON units system registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: Which VFP system register is written.

MASK unsigned int
: The mask indicating updated cumulative bits.

OLD_VALUE unsigned int
: The register's old value overwritten.

VALUE unsigned int
: The new value written to the VFP system register.

VFP_S_REGS

VFP/NEON S 32 bit register write. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ALIASING enum
: Type of register aliasing used.

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

WAYPOINT

Signals end of basic block execution. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: Origin instructions set

IS_COND bool
: Indicates if this is a conditional waypoint.

PC unsigned int
: Origin address (or 0 if unavailable)

TAKEN bool
: Indicates if this waypoint was taken.

TARGET unsigned int
: Destination address

TARGET_ISET enum
: Destination instructions set

WFE_END

WFE ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFE.

WFE_EVENT_REGISTER

WFE event register status: set/clear, reason. Fields:

INST_COUNT unsigned int
: Ticks count.

REASON enum
: Reason for set/clear. Only REASON==0 (Cleared by WFE) clears the bit, all other reasons set it.

WFE_IGNORED

WFE ignored. Fields:

EVENT bool
: This WFE was ignored because the event register was set.

INST_COUNT unsigned int
: Ticks count when ignoring WFE.

TRAPPED bool
: This WFE was trapped.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

WFE_START

WFE entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFE.

WFI_END

WFI ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFI.

WFI_IGNORED

WFI ignored. Fields:

DISABLED bool
: This WFI was ignored because WFI is disabled.

INST_COUNT unsigned int
: Ticks count when ignoring WFI.

REASON enum
: specifies reason why WFI trace was ignored

TRAPPED bool
: This WFI was trapped.

WFI_START

WFI entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFI.

WFI_WAKEUP

WFI wakeup. Fields:

INST_COUNT unsigned int
: Ticks count when WFI wakeup occurred.

REASON enum
: Reason for wakeup.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.39 ARM_Cortex-R5 - trace

This section describes the trace sources.

ASYNC_MEMORY_FAULT

Context ID Register write. Fields:

FAULT unsigned int
: Fault status

PADDR unsigned int
: Physical Address (or 0 if unavailable)

ATOMIC_END_ACCESS

Bus trace access for atomics. Fields:

ACCESS_FAIL bool
: Memory access failed

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

LOAD_VALUE unsigned int
: Loaded values

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ATOMIC_START_ACCESS

Bus trace access for atomics. Fields:

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ArchMsg.Error.exit_code

Fields:

EXITCODE signed int
: Exit code to be returned

KIND string
: Component kind that invoked the exit code trace

REASON string
: Reason why the exit code trace was invoked (optional)

ArchMsg.Warning.TarmacReconstructor

Unexpected event happened during Tarmac reconstruction. Fields:

REASON string
: Reason

ArchMsg.Warning.cache_contents_unknown

A cache has been activated without being invalidated and may contain UNKNOWN content. Fields:

REGIME unsigned int
: Current translation regime

SIDE unsigned int
: 0 d-side cache or unified cache, 1 i-side

ArchMsg.Warning.recursive_branch

An instruction is performing a branch that targets the same instruction. If this was intended then a WFI might be more effective. Fields:

PC unsigned int
: Address of instruction causing the branch

ArchMsg.Warning.recursive_exception

An instruction has generated an exception that targets the same instruction. Fields:

CPSR unsigned int
: Processor state

PC unsigned int
: Address of instruction causing the exception

TYPE enum
: Type of exception

ArchMsg.Warning.secure_vector_fetch_from_nonsecure

Secure vector table is being fetched from Non-secure memory .

BRANCH_MISPREDICT

Simulating branch mispredict. Fields:

PC unsigned int
: Origin address (or 0 if unavailable)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

BRA_DIR

Direct branches, to immediate address. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR

Indirect branches, perhaps to a register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR_CCFAIL

Indirect branches fail, perhaps to a register.

BRA_RET_CCFAIL

Branche return fail, perhaps to a register.

CACHE_MAINTENANCE_OP

Cache Maintenance Operation. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Specified MVA or Set/way

FUNCTION enum
: Clean / Invalidate

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SCOPE enum
: Affected region

SIDE enum
: Inst / Data.

CCFAIL

Conditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCFAIL_UNC

Unconditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS

Conditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS_UNC

Unconditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CODE_CACHE_MAINT

Code cache maintenance. Fields:

END_INCL unsigned int
: Inclusive end addres for by VA or PA requests

NS bool
: Non-secure world for PA

START unsigned int
: Start address for by VA or PA requests

TYPE enum
: Request type.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

COMPILE_BLOCK_END

Last instruction of basic block translated. Fields:

VADDR unsigned int
: Address of next instruction after this basic block

COMPILE_BLOCK_START

First instruction of basic block translated. Fields:

VADDR unsigned int
: Address of first instruction in basic block

COMPILE_INST

ARM instruction compiled. Fields:

DISASS string
: Disassembly of the instruction.

ISET enum
: The instruction set of this instruction.

ITSTATE unsigned int
: The ITSTATE current for the instruction.

OPCODE unsigned int
: The opcode of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

CONTEXTIDR

Context ID Register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

NS bool
: Secure or nonsecure banked register is accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

CONTEXT_SYNC

Called for every context synchronization event. Fields:

ARCH_CSE bool
: Architectural Context Synchronization Event

SYSREG_SYNC bool
: System register synchronization event

CORE_ENDIAN

Core BE8 Big-Endian state changed. Fields:

BE8 bool
: Core BE8 Big-Endian state

CORE_INFO

Static processor attributes. Only triggered by a call to DumpState(). Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ARCH_PROFILE enum
: The architecture profile of the core.

CLUSTER_ID unsigned int
: The cluster ID of this processor.

CORE_NUM unsigned int
: The number of this core in an MP processor.

DETAILED_CORE_FEATURES enum
: Detailed core features as a bitfield. Enum values are bitmasks to extract the info from the field

FPU_VERSION enum
: The VFP version implemented by the core.

MEM_ARCH enum
: The memory architecture of the core.

MIDR_EL1 unsigned int
: MIDR_EL1 value

NUM_CORES unsigned int
: The number of cores in this MP processor.

QUANTUM_SIZE unsigned int
: The default quantum size of the core.

SECURITY_FEATURES bool
: Does the core have security features?

CORE_LOADS

Processor load accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CORE_REGS

Changes of the core registers R0 to R14. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number, 0 to 14.

MODE enum
: Bank of the register accessed.

OLD_VALUE unsigned int
: The old value overwritten.

PHYS_ID enum
: The physical register accessed.

VALUE unsigned int
: The new value written to the register.

CORE_STORES

Processor store accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CP14_READ

Reads to the System Coprocessor registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP14_WRITE

Writes to the System Coprocessor registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CP15_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

CP15_WRITE

System Control Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

CPSR

CPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

OLD_VALUE unsigned int
: The old CPSR value

UNKNOWN unsigned int
: Bits within the register that have unknown value.

VALUE unsigned int
: The new CPSR value

CRYPTO_SPEC

Every crypto instruction speculatively executed.

DATA_CACHE_SET_ALLOC_TAGS

Set allocation tags from DC GVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of memory in bytes.

TAG unsigned int
: The value of the tag being set.

VADDR unsigned int
: Virtual address used for setting allocation tags.

DATA_CACHE_ZERO

Zero and invalidate cache line from DC ZVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of zero'd memory in bytes.

VADDR unsigned int
: Virtual address of the zero'd cache line.

DTCM_ACCESS

Trace TCM accesses.

DTCM_ACCESS_RAZ_WI

Trace TCM accesses. Fields:

ADDR unsigned int
: Address of the RAZ/WI access.

END_COMPILE

Compilation end. Fields:

END_OF_PAGE_COUNT unsigned int
: Number of basic blocks exited due to page boundary since START_COMPILE.

FETCHFAIL_COUNT unsigned int
: Number of basic blocks exited due to ifetch failure START_COMPILE.

INST_COUNT unsigned int
: Number of instructions compiled since START_COMPILE.

NONSEQ_COUNT unsigned int
: Number of basic blocks exited due to non-sequential instructions since START_COMPILE.

PAGE_STRADDLE_COUNT unsigned int
: Number of basic blocks exited due to unaligned instructions crossing page since START_COMPILE.

EXCEPTION

Exception taken. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

IS_PHYSICAL bool
: Physical or Virtual exception

LR unsigned int
: The value assigned to the link register.

PC unsigned int
: The location where the exception occurred.

PREFERRED_RETURN unsigned int
: The preferred return address for the exception.

TARGET_ISET enum
: The instruction set of the exception handler code.

TARGET_PC unsigned int
: The address the exception branches to.

VECTOR enum
: The exception vector.

VECTOR_OFFSET unsigned int
: The offset of an exception vector from the base address.

EXCEPTION_END

Every exception completed.

EXCEPTION_RAISE

Every exception raised.

EXCEPTION_RETURN

Branches on leaving exception. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

EXCEPTION_RETURN_PREBRANCH

Exception return event that must occur before PC and processor state have updated.

EXCEPTION_START

Every exception started.

FP_STATE

Floating point state. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

FPSR unsigned int
: FPSR register value.

MASK unsigned int
: The mask indicating updated cumulative bits.

HLT

HLT instruction occurred. Fields:

IMM16 unsigned int
: The 16-bit immediate value encoded in HLT instruction.

INST

Every instruction executed. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DEBUG_STATE bool
: The instruction is executed in debug state.

DISASS string
: Disassembly of instruction.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The current instruction set.

ITSTATE unsigned int
: The current ITSTATE.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

MODE enum
: The mode the core is in.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

OPCODE unsigned int
: The opcode of the instruction.

PADDR unsigned int
: The physical address of the instruction.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

INST_END

Every instruction completed.

INST_START

Every instruction started. Fields:

ISET enum
: The current instruction set.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

INST_STRADDLE

Instruction straddles boundary. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

ISB

ISB instruction occurred.

ITCM_ACCESS

Trace TCM accesses.

ITCM_ACCESS_RAZ_WI

Trace TCM accesses. Fields:

ADDR unsigned int
: Address of the RAZ/WI access.

LOAD_MULTIPLE_REGS_LIST

List of destination registers (not source) of a multiple registers load instruction. Fields:

REG string
: Register to add into the list

LOCAL_MONITOR

Local monitor activity. Fields:

PADDR unsigned int
: Local Monitor Address

State enum
: State of the monitor (Open/Exclusive)

MODE_CHANGE

Mode change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: The new mode.

OLD_MODE enum
: The old mode.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MPU_TRANS

Address translation information. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

EXEC_PERM enum
: Execution Permission

INNERCACHE_RA bool
: Is the inner cache allocate on read

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

MEMTYPE enum
: Memory type.

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

PADDR unsigned int
: Address of the access.

PAGESIZE unsigned int
: Page size as log2(size).

READ_PERM enum
: Read Permission

REG_NUM signed int
: MPU region-number 0-15 (else 0xFF=none, 0xBB/0xDD=background/default)

SH enum
: Shareability

SIDE enum
: Inst / Data.

WRITE_PERM enum
: Write Permission

PERIODIC

Called for every quantum. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

PC unsigned int
: The address of the next instruction to be executed on this CPU.

PRELOAD_DATA

Data preload from PLD instruction. Fields:

VADDR unsigned int
: Virtual address of the data that should be preloaded.

PRELOAD_INST

Instruction preload from PLI instruction. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VADDR unsigned int
Virtual address of the instruction that should be preloaded.

PRE_CORE_LOAD

Trace just before a core load. Fields:

CURRENT_TIME unsigned int
The core's current time, as simulated time plus local time.
LOCAL_TIME unsigned int
The core's local time, relative to the current quantum.

PRE_CORE_STORE

Trace just before a core store. Fields:

CURRENT_TIME unsigned int
The core's current time, as simulated time plus local time.
LOCAL_TIME unsigned int
The core's local time, relative to the current quantum.

PSTATE_PACTIVE_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Operating mode after the transition
OLD_STATE enum
Operating mode before the transition

PSTATE_PACTIVE_OPMODE_TRANSITION_DENIED

Trace the denied pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Operating mode after the transition
OLD_STATE enum
Operating mode before the transition
REASON string
Reason for the transition being denied

PSTATE_PACTIVE_POWER_TRANSITION

Trace pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Power mode after the transition
OLD_STATE enum
Power mode before the transition

PSTATE_PACTIVE_POWER_TRANSITION_DENIED

Trace the denied pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_THREAD_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions for threads on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

RUN_STATE

Run state transition. Fields:

INST_COUNT unsigned int
: Ticks count at point of transition.

NEW enum
: New run state.

OLD enum
: Old run state.

SEMIHOSTING_CALL

Call of a semihost function occurred. Fields:

PC unsigned int
: The program counter after the semihosting call.

SEMIHOSTING_PRECALL

About to call semihost. Fields:

EL2 bool
: Use EL2 translation regime.

NS bool
: Is Non-Secure.

REG_WIDTH unsigned int
: The current register width in bytes.

SIGNAL

External signal state change. Fields:

SIGNAL enum
: Signal that changed

STATE bool
: Signal asserted state

SPSR

SPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MODE enum
: Which of the banked SPSR registers is written.

OLD_VALUE unsigned int
: The old SPSR value

VALUE unsigned int
: The new SPSR value

START_COMPILE

Compilation started. Fields:

VADDR unsigned int
: Instruction where compilation begins.

STG_MULTIPLE_MICRO_OPS_ADDRESS_RANGE

Address range for MTE stores with multiple uops. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

STORE_MULTIPLE_REGS_ADDRESS_RANGE

Address range for stores with multiple registers. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

SUBOPTIMAL_LDST_RETIRED

suboptimal load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1: half-word, 2: word ...)

OPTIMAL_ALIGN_SELECTOR unsigned int
: Selector to decide the optimal alignment size (1: access width, 2: 4-bytes, 3: 8-bytes, ..., 12: 4-kbytes)

VADDR unsigned int
: The virtual address of the access

SYNC

Called for every synchronization, that is every quantum break. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

LOCAL_QUANTUM unsigned int
: The local quantum of this CPU.

LOCAL_TIME unsigned int
: The local time of this CPU.

SYNC_REASON enum
: Reason for sync.

SYSCALL

System call instruction executed. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

IMM unsigned int
: Immediate value of the system call instruction.

TYPE enum
: System call type.

VADDR unsigned int
: Instruction that caused the system call.

SYSREG_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE32

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SYSREG_UPDATE64

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

UNALIGNED_LDST_RETIRED

Processor unaligned load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1=>half-word, 2->word ...)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NUMBER_OF_BEATS unsigned int
: Number of accesses (beats) in this burst (Total size in bytes is ACCESS_SIZE * NUMBER_OF_BEATS)

VADDR unsigned int
: The virtual address of the access

VFP_D_REGS

VFP/NEON D 64 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_Q_REGS

VFP/NEON Q 128 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding byte in the register. For example, 0x1 would indicate a write to VALUE[7:0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_SYS_REGS

Writes to the VFP/NEON units system registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: Which VFP system register is written.

MASK unsigned int
: The mask indicating updated cumulative bits.

OLD_VALUE unsigned int
: The register's old value overwritten.

VALUE unsigned int
: The new value written to the VFP system register.

VFP_S_REGS

VFP/NEON S 32 bit register write. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ALIASING enum
: Type of register aliasing used.

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

WAYPOINT

Signals end of basic block execution. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: Origin instructions set

IS_COND bool
: Indicates if this is a conditional waypoint.

PC unsigned int
: Origin address (or 0 if unavailable)

TAKEN bool
: Indicates if this waypoint was taken.

TARGET unsigned int
: Destination address

TARGET_ISET enum
: Destination instructions set

WFE_END

WFE ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFE.

WFE_EVENT_REGISTER

WFE event register status: set/clear, reason. Fields:

INST_COUNT unsigned int
: Ticks count.

REASON enum
: Reason for set/clear. Only REASON==0 (Cleared by WFE) clears the bit, all other reasons set it.

WFE_IGNORED

WFE ignored. Fields:

EVENT bool
: This WFE was ignored because the event register was set.

INST_COUNT unsigned int
: Ticks count when ignoring WFE.

TRAPPED bool
: This WFE was trapped.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

WFE_START

WFE entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFE.

WFI_END

WFI ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFI.

WFI_IGNORED

WFI ignored. Fields:

DISABLED bool
: This WFI was ignored because WFI is disabled.

INST_COUNT unsigned int
: Ticks count when ignoring WFI.

REASON enum
: specifies reason why WFI trace was ignored

TRAPPED bool
: This WFI was trapped.

WFI_START

WFI entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFI.

WFI_WAKEUP

WFI wakeup. Fields:

INST_COUNT unsigned int
: Ticks count when WFI wakeup occurred.

REASON enum
: Reason for wakeup.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.40 ARM_Cortex-R7 - trace

This section describes the trace sources.

ASYNC_MEMORY_FAULT

Context ID Register write. Fields:

FAULT unsigned int
: Fault status

PADDR unsigned int
: Physical Address (or 0 if unavailable)

ATOMIC_END_ACCESS

Bus trace access for atomics. Fields:

ACCESS_FAIL bool
: Memory access failed

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

LOAD_VALUE unsigned int
: Loaded values

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ATOMIC_START_ACCESS

Bus trace access for atomics. Fields:

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ArchMsg.Error.exit_code

Fields:

EXITCODE signed int
: Exit code to be returned

KIND string
: Component kind that invoked the exit code trace

REASON string
: Reason why the exit code trace was invoked (optional)

ArchMsg.Warning.IgnoredWarmResetRequest

Event fires when software attempts to request warm reset through write to deprecated DBGPRCR.CWRR.

ArchMsg.Warning.TarmacReconstructor

Unexpected event happened during Tarmac reconstruction. Fields:

REASON string
: Reason

ArchMsg.Warning.Unpredictably Indexed PM Event Register

Fields:

IsDirect unsigned int
: Direct Access rather than indirect

N unsigned int
: Number of Accessible Registers

SEL unsigned int
: Selector

ArchMsg.Warning.branch_to_unaligned_address

Fields:

ADDR unsigned int
: Unaligned branch address

ArchMsg.Warning.cache_contents_unknown

A cache has been activated without being invalidated and may contain UNKNOWN content. Fields:

REGIME unsigned int
: Current translation regime

SIDE unsigned int
: 0 d-side cache or unified cache, 1 i-side

ArchMsg.Warning.dcimvac_matches_watchpoint

Watchpoints matching an AArch32 DCIMVA are implementation defined.

ArchMsg.Warning.decode_fieldmismatch

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

FIELD1 enum
Field name
FIELD2 enum
Field name

ArchMsg.Warning.decode_fracbitsnegative

VCVT Instruction is unpredictable with a negative number of fraction bits.

ArchMsg.Warning.decode_initblock

Instruction is unpredictable in an IT block.

ArchMsg.Warning.decode_initblocknotlast

Instruction is unpredictable in an IT block when not the last.

ArchMsg.Warning.decode_invalid_condition

Instruction is unpredictable when the condition is not AL.

ArchMsg.Warning.decode_invalidfieldcombination

Fields:

FIELD1 enum
Field name
FIELD2 enum
Field name

ArchMsg.Warning.decode_invalidvalue

Fields:

FIELD enum
Field name

ArchMsg.Warning.decode_mem_hint_unallocated

There is no memory hint allocated to this bit pattern.

ArchMsg.Warning.decode_registermatch

Fields:

FIELD1 enum
Field name
FIELD2 enum
Field name

ArchMsg.Warning.decode_registermismatch

Fields:

FIELD1 enum
Field name
FIELD2 enum
Field name

ArchMsg.Warning.decode_registeroutofrange

Float point instruction is unpredictable indexing beyond the end of the register bank.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_sbzsbo

Reserved bits in the instruction are not canonical.

ArchMsg.Warning.decode_transactiontoobig

Attempt to perform a memory transaction of more than 128 bytes.

ArchMsg.Warning.decode_unpred_other

Instruction is unpredictable.

ArchMsg.Warning.decode_unpreduseofpc

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13inreglist

Instruction is unpredictable with R13 in register-list.

ArchMsg.Warning.decode_useofd16orhigher

Attempt to use register D16 or higher where it is not allowed.

ArchMsg.Warning.decode_writebackandbaseinlist

Instruction is unpredictable with write back when the base register is in the transfer list.

ArchMsg.Warning.decode_zeroregistersinlist

Instruction is unpredictable when no registers are to be transfered.

ArchMsg.Warning.recursive_branch

An instruction is performing a branch that targets the same instruction. If this was intended then a WFI might be more effective. Fields:

PC unsigned int
: Address of instruction causing the branch

ArchMsg.Warning.recursive_exception

An instruction has generated an exception that targets the same instruction. Fields:

CPSR unsigned int
: Processor state

PC unsigned int
: Address of instruction causing the exception

TYPE enum
: Type of exception

ArchMsg.Warning.reentrant_vector_catch

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR unsigned int
vector address

ArchMsg.Warning.reserved_it_state

Fields:

ITSTATE unsigned int
reserved IT state

ArchMsg.Warning.secure_vector_fetch_from_nonsecure

Secure vector table is being fetched from Non-secure memory .

ArchMsg.Warning.unpredictable_a32_breakpoint

Fields:

ADDR1 unsigned int
Address
ADDR2 unsigned int
Breakpoint address
BAS unsigned int
BAS Field
IS_ADDRESS_MISMATCH_BREAKPOINT bool
Is Mismatch breakpoint, not a match breakpoint
N unsigned int
Breakpoint number
OFFSET unsigned int
Indicates which halfword actually considering into the A32 instruction

ArchMsg.Warning.unpredictable_banked_register_access

Fields:

R unsigned int
R bit
SYSM unsigned int
SYSm

ArchMsg.Warning.unpredictable_ccfail_undef

Fields:

OPCODE unsigned int
opcode of the instruction
VADDR unsigned int
Virtual address of the instruction being executed

ArchMsg.Warning.unpredictable_ici_bits

ICI bits do not apply or are architecturally unpredictable for instruction.

ArchMsg.Warning.unpredictable_t32_breakpoint

Fields:

ADDR1 unsigned int
Address
ADDR2 unsigned int
Breakpoint address
BAS unsigned int
BAS Field

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

IS_ADDRESS_MISMATCH_BREAKPOINT bool
: Is Mismatch breakpoint, not a match breakpoint

N unsigned int
: Breakpoint number

OFFSET unsigned int
: Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_t32_vector_catch

Fields:

ADDR unsigned int
: Address

OFFSET unsigned int
: Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_watchpoint_far

Fields:

LOWER_BOUND unsigned int
: the lowest address accessed by the instruction that triggered the watchpoint

UPPER_BOUND unsigned int
: the highest watchpointed address accessed by that instruction

ArchMsg.Warning.warning_AdvSIMDExpandImmUnexpectedZero

AdvSIMDExpandImm may treat this immediate value as UNPREDICTABLE.

ArchMsg.Warning.warning_ConditionalSMC

SMC instruction has UNPREDICTABLE effects when conditional when combined with traps.

ArchMsg.Warning.warning_access_crosses_page_boundary

Access crosses page boundary.

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault_on_both_pages

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_spanning_different_memory

Fields:

ADDR unsigned int
: address of access crossing page

MEMTYPE_PAGE1 signed int
: First Page's memory type (0-Normal, 1-Non-Normal)

MEMTYPE_PAGE2 signed int
: Second Page's memory type (0-Normal, 1-Non-Normal)

ArchMsg.Warning.warning_access_wraps_around_memory

Fields:

WIDTH unsigned int
: Address width

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_bx_from_thumbee

Fields:

ADDR unsigned int
: Destination address

ArchMsg.Warning.warning_ccsidr_unimplemented_level

Fields:

CSSELR unsigned int
: current effective value of CSSELR

ArchMsg.Warning.warning_change_to_ns_when_tge_set

Attempting to change NS to 1 when HCR.TGE = 1.

ArchMsg.Warning.warning_contiguous_bit_check_abort

Fields:

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ArchMsg.Warning.warning_contiguous_bit_error

Fields:

CONTIG_ADDR unsigned int
: address of the TLB first read, with which this entry is expected to be contiguous

CONTIG_DATA unsigned int
: expected contents based on the entry at CONTIG_ADDR

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ENTRY_DATA unsigned int
: contents of conflicting TLB entry

ArchMsg.Warning.warning_cp10_cp11_mismatch

Fields:

EL unsigned int
: EL owning the control

ArchMsg.Warning.warning_cp10_cp11_reserved_value

Fields:

CP10 unsigned int
: CP10 access

CP11 unsigned int
: CP11 access

ArchMsg.Warning.warning_csselr_level_out_of_range

Fields:

IMPLEMENTED unsigned int
: number of cache levels implemented

LEVEL unsigned int
: written value of CSSELR.Level

TYPE enum
: The type of cache selected by CSSELR.InD and TnD

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_debug_flow_control_bits_not_obeyed

Fields:

ERROR enum
: type of error resulting

REGISTER enum
: register under consideration

ArchMsg.Warning.warning_debug_register_access_during_reset

Fields:

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

ArchMsg.Warning.warning_decode_cps_inconsistent_fields

The mode setting fields in the CPS instruction are inconsistent.

ArchMsg.Warning.warning_decode_invalid_state

Instruction is unpredictable in current exception-level/security state.

ArchMsg.Warning.warning_default_cacheable_mmu_on

Default cacheable is enabled (HCR.DC = 1) while MMU is enabled.

ArchMsg.Warning.warning_default_cacheable_vmmu_off

Default cacheable is enabled (HCR.DC = 1) while HCR.VM = 0.

ArchMsg.Warning.warning_deprecated_wvr_bit_2_set

Fields:

N unsigned int
: index

WVR unsigned int
: DBGWVR

ArchMsg.Warning.warning_eret_while_software_step_active_pending

Missing ISB between setting MDSCR_EL1.SS and ERET.

ArchMsg.Warning.warning_exclusive_to_non_normal

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_exclusive_to_non_writeback

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_execute_from_device_memory

An attempt was made to execute from Device memory.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_illegal_cpsr_mode

Fields:

NEW_MODE unsigned int
: New value of CPSR.M

ArchMsg.Warning.warning_illegal_srs_mode

Fields:

MODE unsigned int
: mode for Banked SP

ArchMsg.Warning.warning_implementation_defined_read_debug_register_in_SCS

Read Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_implementation_defined_sequential_security_transitions_supported

The behaviour of a sequential instruction fetches that cross from non-secure to secure memory and contain SG instruction is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_implementation_defined_stack_limit_check_supported

It is IMPLEMENTATION DEFINED whether stack pointer limit checking is performed for the this instructions. Fields:

ADDRESS unsigned int
: Address of instruction

OPCODE unsigned int
: instruction opcode

ArchMsg.Warning.warning_implementation_defined_write_debug_register_in_SCS

Write Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

DATA unsigned int
: data attempted to be written

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_invalid_tcr_granule

Fields:

REQUEST signed int
: page size requested (or 0 for reserved)

SUBSTITUTE signed int
: best guess available page size

TG_ID bool
: bits TG0 or TG1

ArchMsg.Warning.warning_load_multiple_user_registers_from_user_mode

An LDM(user registers) instruction executed from user mode.

ArchMsg.Warning.warning_load_pc_from_unaligned

PC loaded from an unaligned location.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_mte_in_mem_attr_not_supported

Fields:

ATTR unsigned int
: Memory attribute

ArchMsg.Warning.warning_reserved_breakpoint_state_match

Fields:

IS_BREAKPOINT unsigned int
: Is breakpoint, not watchpoint

ArchMsg.Warning.warning_shareability

Unpredictable: combination of the 1st and 2nd stages of translation is Normal Inner Non-Cacheable, Outer Non-Cacheable.

ArchMsg.Warning.warning_software_step_set_while_enabled

MDSCR_EL1.SS set to 1 while software step debug exceptions are enabled.

ArchMsg.Warning.warning_thumb_instruction_wraps_around_memory

Thumb instruction warps around memory.

ArchMsg.Warning.warning_ttbr_sbz_bits_are_not_zero

Fields:

MSB unsigned int
: x-1

TTBR enum
: which TTBR register

ArchMsg.Warning.warning_unaligned_address_dbgdtrrx_write

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_unaligned_address_dbgdtrtx_read

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_unaligned_to_device

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_unaligned_to_strongly_ordered

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_unknown_sau_rnr

SAU_RNR was set to an unsupported value.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_AIRCR_PRIP_and_BFHFNMINP

The effect of setting both AIRCR.BFHFNMINP and AIRCR.PRIP to 1 is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTCLRACTIVE_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTCLRACTIVE if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_and_SYSRESETREQ

When the processor is halted in Debug state, if a write to the register writes a 1 to both VECTRESET and SYSRESETREQ, the behavior is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTRESET if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_incorrect_VKEY

The value 0x05FA must be written to AIRCR.VKEY, otherwise the register write is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_EXE_RETURN_Reserved_Bit

EXC_RETURN[23:7] are reserved with the special condition that all bits should be written as one. Values other than all 1s are UNPREDICTABLE. Fields:

EXC_RETURN `unsigned int`
: Special PC value

SBO_MASK `unsigned int`
: Which bits Should Be One to not UNPRED

ArchMsg.Warning.warning_unpredictable_change_to_DEMCR_MON_STEP_at_insufficient_priority

The effect of changing DEMCR.MON_STEP at an execution priority that is lower than the priority of the DebugMonitor exception is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_change_to_priority_of_active_exception

Changing the priority of an active exception is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_clear_lspact

Fields:

DRVEXC_TAKEN `unsigned int`
: derived exception taken not pended

REACHED_LAST_STORE `unsigned int`
: only the last faulted

ArchMsg.Warning.warning_unpredictable_dbg_exit_unaligned_dlr

Debug exit to AArch32 with DLR[0] set to 1. PC[0] can be set to 0 or to DLR[0].

ArchMsg.Warning.warning_unpredictable_exception_catch

Generation of an exception catch event is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_inconsistent_state

state on exception return is unpredictable.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_exception_return_instruction

exception return instruction is unpredictable.

ArchMsg.Warning.warning_unpredictable_in_debug_state

Instruction is UNPREDICTABLE when executed in debug state.

ArchMsg.Warning.warning_unpredictable_mair_encoding

Fields:

ATTR unsigned int
: Memory attribute

ArchMsg.Warning.warning_unpredictable_pmu_counter_access

Access to PMU counters from non-secure EL0 or EL1 is UNPREDICTABLE with MDCR_EL2.HPMN set to 0.

ArchMsg.Warning.warning_unpredictable_prioritization_breakpoint_match_vector_catch

Two events with the same priority occurred at the same instruction: Address Matching Vector Catch debug event, and Breakpoint debug event It is constrained unpredictable which is taken.

ArchMsg.Warning.warning_unpredictable_stack_selection

UNPREDICTABLE state on context switch.

ArchMsg.Warning.warning_unpredictable_tsize_out_of_range

Fields:

TCR enum
: Which TCR

TSIZE signed int
: Value of TSize

TSIZE_MAX signed int
: The maximum allowed value

TTBR unsigned int
: Which TTBR region

ArchMsg.Warning.warning_unpredictable_unaligned_pc_as_base_register

The PC value must be word-aligned, otherwise the behavior of the instruction is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_unaligned_pop_stack

Fields:

FRAME_ADDR unsigned int
: unaligned stack frame address

ArchMsg.Warning.warning_unpredictable_vmsa_memattrib

Fields:

TEXCB unsigned int
: texcb value

TEX_REMAP bool
: use tex remap

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_vtcr_t0sz_sl0

Fields:

SL0 signed int
: VTCR.SL0, starting level for stage 2 translation table walks

T0SZ signed int
: VTCR.T0SZ, required input address range

ArchMsg.Warning.warning_unpredictable_write_DHCSR

Fields:

NEW unsigned int
: data attempted to be written

OLD unsigned int
: previous value

ArchMsg.Warning.warning_unpredictable_write_SHCSR

UNPREDICTABLE setting a pending bit to 1 for an exception with priority >= execution priority.

ArchMsg.Warning.warning_unsupported_access_to_memory_mapped_register

Access to memory mapped register is unsupported.

ArchMsg.Warning.warning_user_jmcr_access

Fields:

WRITE unsigned int
: access is write

ArchMsg.Warning.warning_wcr_mask_and_bas

DBGWCR_EL1.MASK is non-zero and BAS is not 0xff.

ArchMsg.Warning.warning_wcr_mask_reserved

DBGWCR_ELn.MASK is set to a reserved value.

ArchMsg.Warning.warning_wcr_non_configuous_bas

DBGWCR_EL1.BAS has non-contiguous set of ones.

ArchMsg.Warning.warning_wvr_masked_bit_not_zero

DBGWVRn_EL1.VA contains a bit masked by DBGWCRn_EL1.MASK that is not zero.

BRANCH_MISPREDICT

Simulating branch mispredict. Fields:

PC unsigned int
: Origin address (or 0 if unavailable)

BRA_DIR

Direct branches, to immediate address. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR

Indirect branches, perhaps to a register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR_CCFAIL

Indirect branches fail, perhaps to a register.

BRA_RET_CCFAIL

Branche return fail, perhaps to a register.

CACHE_MAINTENANCE_OP

Cache Maintenance Operation. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Specified MVA or Set/way

FUNCTION enum
: Clean / Invalidate

SCOPE enum
: Affected region

SIDE enum
: Inst / Data.

CCFAIL

Conditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCFAIL_UNC

Unconditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS

Conditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS_UNC

Unconditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CODE_CACHE_MAINT

Code cache maintenance. Fields:

END_INCL unsigned int
: Inclusive end addres for by VA or PA requests

NS bool
: Non-secure world for PA

START unsigned int
: Start address for by VA or PA requests

TYPE enum
: Request type.

COMPILE_BLOCK_END

Last instruction of basic block translated. Fields:

VADDR unsigned int
: Address of next instruction after this basic block

COMPILE_BLOCK_START

First instruction of basic block translated. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VADDR unsigned int
: Address of first instruction in basic block

COMPILE_INST

ARM instruction compiled. Fields:

DISASS string
: Disassembly of the instruction.

ISET enum
: The instruction set of this instruction.

ITSTATE unsigned int
: The ITSTATE current for the instruction.

OPCODE unsigned int
: The opcode of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

CONTEXTIDR

Context ID Register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

NS bool
: Secure or nonsecure banked register is accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

CONTEXT_SYNC

Called for every context synchronization event. Fields:

ARCH_CSE bool
: Architectural Context Synchronization Event

SYSREG_SYNC bool
: System register synchronization event

CORE_ENDIAN

Core BE8 Big-Endian state changed. Fields:

BE8 bool
: Core BE8 Big-Endian state

CORE_INFO

Static processor attributes. Only triggered by a call to DumpState(). Fields:

ARCH_PROFILE enum
: The architecture profile of the core.

CLUSTER_ID unsigned int
: The cluster ID of this processor.

CORE_NUM unsigned int
: The number of this core in an MP processor.

DETAILED_CORE_FEATURES enum
: Detailed core features as a bitfield. Enum values are bitmasks to extract the info from the field

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

FPU_VERSION enum
: The VFP version implemented by the core.

MEM_ARCH enum
: The memory architecture of the core.

MIDR_EL1 unsigned int
: MIDR_EL1 value

NUM_CORES unsigned int
: The number of cores in this MP processor.

QUANTUM_SIZE unsigned int
: The default quantum size of the core.

SECURITY_FEATURES bool
: Does the core have security features?

CORE_LOADS

Processor load accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_REGS

Changes of the core registers R0 to R14. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number, 0 to 14.

MODE enum
: Bank of the register accessed.

OLD_VALUE unsigned int
: The old value overwritten.

PHYS_ID enum
: The physical register accessed.

VALUE unsigned int
: The new value written to the register.

CORE_STORES

Processor store accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CP14_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP14_WRITE

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

CP15_WRITE

System Control Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

opc2 unsigned int
: opcode 2

CP15_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

CPSR

CPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

OLD_VALUE unsigned int
: The old CPSR value

UNKNOWN unsigned int
: Bits within the register that have unknown value.

VALUE unsigned int
: The new CPSR value

CRYPTO_SPEC

Every crypto instruction speculatively executed.

CorePowerStateChange

Core power-up state has changed. Fields:

PU bool
: Whether the core powered up

DATA_CACHE_SET_ALLOC_TAGS

Set allocation tags from DC GVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of memory in bytes.

TAG unsigned int
: The value of the tag being set.

VADDR unsigned int
: Virtual address used for setting allocation tags.

DATA_CACHE_ZERO

Zero and invalidate cache line from DC ZVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of zero'd memory in bytes.

VADDR unsigned int
: Virtual address of the zero'd cache line.

DEBUG_EVENT

Hardware debug support event. Fields:

EVENT enum
: Description of event

VALUE unsigned int
: data value

DTCM_ACCESS

Trace TCM accesses.

DTCM_ACCESS_RAZ_WI

Trace TCM accesses. Fields:

ADDR unsigned int
: Address of the RAZ/WI access.

END_COMPILE

Compilation end. Fields:

END_OF_PAGE_COUNT unsigned int
: Number of basic blocks exited due to page boundary since START_COMPILE.

FETCHFAIL_COUNT unsigned int
: Number of basic blocks exited due to ifetch failure START_COMPILE.

INST_COUNT unsigned int
: Number of instructions compiled since START_COMPILE.

NONSEQ_COUNT unsigned int
: Number of basic blocks exited due to non-sequential instructions since START_COMPILE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PAGE_STRADDLE_COUNT unsigned int
: Number of basic blocks exited due to unaligned instructions crossing page since START_COMPILE.

EXCEPTION

Exception taken. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

IS_PHYSICAL bool
: Physical or Virtual exception

LR unsigned int
: The value assigned to the link register.

PC unsigned int
: The location where the exception occurred.

PREFERRED_RETURN unsigned int
: The preferred return address for the exception.

TARGET_ISET enum
: The instruction set of the exception handler code.

TARGET_PC unsigned int
: The address the exception branches to.

VECTOR enum
: The exception vector.

VECTOR_OFFSET unsigned int
: The offset of an exception vector from the base address.

EXCEPTION_END

Every exception completed.

EXCEPTION_RAISE

Every exception raised.

EXCEPTION_RETURN

Branches on leaving exception. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

EXCEPTION_RETURN_PREBRANCH

Exception return event that must occur before PC and processor state have updated.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

EXCEPTION_START

Every exception started.

FP_STATE

Floating point state. Fields:

FPSR unsigned int
: FPSR register value.

MASK unsigned int
: The mask indicating updated cumulative bits.

HLT

HLT instruction occurred. Fields:

IMM16 unsigned int
: The 16-bit immediate value encoded in HLT instruction.

INST

Every instruction executed. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DEBUG_STATE bool
: The instruction is executed in debug state.

DISASS string
: Disassembly of instruction.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The current instruction set.

ITSTATE unsigned int
: The current ITSTATE.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

MODE enum
: The mode the core is in.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

OPCODE unsigned int
: The opcode of the instruction.

PADDR unsigned int
: The physical address of the instruction.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

INST_END

Every instruction completed.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_START

Every instruction started. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

INST_STRADDLE

Instruction straddles boundary. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

ISB

ISB instruction occurred.

ITCM_ACCESS

Trace TCM accesses.

ITCM_ACCESS_RAZ_WI

Trace TCM accesses. Fields:

ADDR unsigned int
: Address of the RAZ/WI access.

LOAD_MULTIPLE_REGS_LIST

List of destination registers (not source) of a multiple registers load instruction. Fields:

REG string
: Register to add into the list

LOCAL_MONITOR

Local monitor activity. Fields:

PADDR unsigned int
: Local Monitor Address

State enum
: State of the monitor (Open/Exclusive)

MEMMAP_DEBUG_READ

Memory mapped reads to the debug registers. Fields:

ADDR unsigned int
: address

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

MEMMAP_DEBUG_WRITE

Memory mapped writes to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

MODE_CHANGE

Mode change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: The new mode.

OLD_MODE enum
: The old mode.

MPU_TRANS

Address translation information. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

EXEC_PERM enum
: Execution Permission

INNERCACHE_RA bool
: Is the inner cache allocate on read

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

MEMTYPE enum
: Memory type.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

PADDR unsigned int
: Address of the access.

PAGESIZE unsigned int
: Page size as log2(size).

READ_PERM enum
: Read Permission

REG_NUM signed int
: MPU region-number 0-15 (else 0xFF=none, 0xBB/0xDD=background/default)

SH enum
: Shareability

SIDE enum
: Inst / Data.

WRITE_PERM enum
: Write Permission

PERIODIC

Called for every quantum. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

PC unsigned int
: The address of the next instruction to be executed on this CPU.

PMU_COUNTER_OVERFLOW

PMU counter overflow. Fields:

EVENT_ID unsigned int
: Event Identifier

INDEX unsigned int
: Counter index (as selected by PMSELR).

INTERRUPT bool
: Is interrupt enabled on overflow for this counter

PRELOAD_DATA

Data preload from PLD instruction. Fields:

VADDR unsigned int
: Virtual address of the data that should be preloaded.

PRELOAD_INST

Instruction preload from PLI instruction. Fields:

VADDR unsigned int
: Virtual address of the instruction that should be preloaded.

PRE_CORE_LOAD

Trace just before a core load. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_CORE_STORE

Trace just before a core store. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PSTATE_PACTIVE_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

PSTATE_PACTIVE_OPMODE_TRANSITION_DENIED

Trace the denied pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_POWER_TRANSITION

Trace pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

PSTATE_PACTIVE_POWER_TRANSITION_DENIED

Trace the denied pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

REASON string
: Reason for the transition being denied

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PSTATE_PACTIVE_THREAD_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions for threads on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

RUN_STATE

Run state transition. Fields:

INST_COUNT unsigned int
: Ticks count at point of transition.

NEW enum
: New run state.

OLD enum
: Old run state.

SEMIHOSTING_CALL

Call of a semihost function occurred. Fields:

PC unsigned int
: The program counter after the semihosting call.

SEMIHOSTING_PRECALL

About to call semihost. Fields:

EL2 bool
: Use EL2 translation regime.

NS bool
: Is Non-Secure.

REG_WIDTH unsigned int
: The current register width in bytes.

SIGNAL

External signal state change. Fields:

SIGNAL enum
: Signal that changed

STATE bool
: Signal asserted state

SPSR

SPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: Which of the banked SPSR registers is written.

OLD_VALUE unsigned int
: The old SPSR value

VALUE unsigned int
: The new SPSR value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

START_COMPILE

Compilation started. Fields:

VADDR unsigned int
: Instruction where compilation begins.

STG_MULTIPLE_MICRO_OPS_ADDRESS_RANGE

Address range for MTE stores with multiple uops. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

STORE_MULTIPLE_REGS_ADDRESS_RANGE

Address range for stores with multiple registers. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

SUBOPTIMAL_LDST_RETIRED

suboptimal load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1: half-word, 2: word ...)

OPTIMAL_ALIGN_SELECTOR unsigned int
: Selector to decide the optimal alignment size (1: access width, 2: 4-bytes, 3: 8-bytes, ..., 12: 4-kbytes)

VADDR unsigned int
: The virtual address of the access

SYNC

Called for every synchronization, that is every quantum break. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

LOCAL_QUANTUM unsigned int
: The local quantum of this CPU.

LOCAL_TIME unsigned int
: The local time of this CPU.

SYNC_REASON enum
: Reason for sync.

SYSCALL

System call instruction executed. Fields:

IMM unsigned int
: Immediate value of the system call instruction.

TYPE enum
: System call type.

VADDR unsigned int
: Instruction that caused the system call.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SYSREG_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE32

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE64

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

UNALIGNED_LDST_RETIRED

Processor unaligned load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1=>half-word, 2->word ...)

NUMBER_OF_BEATS unsigned int
: Number of accesses (beats) in this burst (Total size in bytes is ACCESS_SIZE * NUMBER_OF_BEATS)

VADDR unsigned int
: The virtual address of the access

VFP_D_REGS

VFP/NEON D 64 bit register write. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_Q_REGS

VFP/NEON Q 128 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding byte in the register. For example, 0x1 would indicate a write to VALUE[7:0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_SYS_REGS

Writes to the VFP/NEON units system registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: Which VFP system register is written.

MASK unsigned int
: The mask indicating updated cumulative bits.

OLD_VALUE unsigned int
: The register's old value overwritten.

VALUE unsigned int
: The new value written to the VFP system register.

VFP_S_REGS

VFP/NEON S 32 bit register write. Fields:

ALIASING enum
: Type of register aliasing used.

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

WAYPOINT

Signals end of basic block execution. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: Origin instructions set

IS_COND bool
: Indicates if this is a conditional waypoint.

PC unsigned int
: Origin address (or 0 if unavailable)

TAKEN bool
: Indicates if this waypoint was taken.

TARGET unsigned int
: Destination address

TARGET_ISET enum
: Destination instructions set

WFE_END

WFE ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFE.

WFE_EVENT_REGISTER

WFE event register status: set/clear, reason. Fields:

INST_COUNT unsigned int
: Ticks count.

REASON enum
: Reason for set/clear. Only REASON==0 (Cleared by WFE) clears the bit, all other reasons set it.

WFE_IGNORED

WFE ignored. Fields:

EVENT bool
: This WFE was ignored because the event register was set.

INST_COUNT unsigned int
: Ticks count when ignoring WFE.

TRAPPED bool
: This WFE was trapped.

WFE_START

WFE entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

WFI_END

WFI ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFI.

WFI_IGNORED

WFI ignored. Fields:

DISABLED bool
: This WFI was ignored because WFI is disabled.

INST_COUNT unsigned int
: Ticks count when ignoring WFI.

REASON enum
: specifies reason why WFI trace was ignored

TRAPPED bool
: This WFI was trapped.

WFI_START

WFI entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFI.

WFI_WAKEUP

WFI wakeup. Fields:

INST_COUNT unsigned int
: Ticks count when WFI wakeup occurred.

REASON enum
: Reason for wakeup.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.41 ARM_Cortex-R8 - trace

This section describes the trace sources.

ASYNC_MEMORY_FAULT

Context ID Register write. Fields:

FAULT unsigned int
: Fault status

PADDR unsigned int
: Physical Address (or 0 if unavailable)

ATOMIC_END_ACCESS

Bus trace access for atomics. Fields:

ACCESS_FAIL bool
: Memory access failed

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

LOAD_VALUE unsigned int
: Loaded values

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ATOMIC_START_ACCESS

Bus trace access for atomics. Fields:

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ArchMsg.Error.exit_code

Fields:

EXITCODE signed int
: Exit code to be returned

KIND string
: Component kind that invoked the exit code trace

REASON string
: Reason why the exit code trace was invoked (optional)

ArchMsg.Warning.IgnoredWarmResetRequest

Event fires when software attempts to request warm reset through write to deprecated DBGPRCR.CWRR.

ArchMsg.Warning.TarmacReconstructor

Unexpected event happened during Tarmac reconstruction. Fields:

REASON string
: Reason

ArchMsg.Warning.Unpredictably Indexed PM Event Register

Fields:

IsDirect unsigned int
: Direct Access rather than indirect

N unsigned int
: Number of Accessible Registers

SEL unsigned int
: Selector

ArchMsg.Warning.branch_to_unaligned_address

Fields:

ADDR unsigned int
: Unaligned branch address

ArchMsg.Warning.cache_contents_unknown

A cache has been activated without being invalidated and may contain UNKNOWN content. Fields:

REGIME unsigned int
: Current translation regime

SIDE unsigned int
: 0 d-side cache or unified cache, 1 i-side

ArchMsg.Warning.dcimvac_matches_watchpoint

Watchpoints matching an AArch32 DCIMVA are implementation defined.

ArchMsg.Warning.decode_fieldmismatch

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

FIELD1 enum
Field name
FIELD2 enum
Field name

ArchMsg.Warning.decode_fracbitsnegative

VCVT Instruction is unpredictable with a negative number of fraction bits.

ArchMsg.Warning.decode_initblock

Instruction is unpredictable in an IT block.

ArchMsg.Warning.decode_initblocknotlast

Instruction is unpredictable in an IT block when not the last.

ArchMsg.Warning.decode_invalid_condition

Instruction is unpredictable when the condition is not AL.

ArchMsg.Warning.decode_invalidfieldcombination

Fields:

FIELD1 enum
Field name
FIELD2 enum
Field name

ArchMsg.Warning.decode_invalidvalue

Fields:

FIELD enum
Field name

ArchMsg.Warning.decode_mem_hint_unallocated

There is no memory hint allocated to this bit pattern.

ArchMsg.Warning.decode_registermatch

Fields:

FIELD1 enum
Field name
FIELD2 enum
Field name

ArchMsg.Warning.decode_registermismatch

Fields:

FIELD1 enum
Field name
FIELD2 enum
Field name

ArchMsg.Warning.decode_registeroutofrange

Float point instruction is unpredictable indexing beyond the end of the register bank.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_sbzsbo

Reserved bits in the instruction are not canonical.

ArchMsg.Warning.decode_transactiontoobig

Attempt to perform a memory transaction of more than 128 bytes.

ArchMsg.Warning.decode_unpred_other

Instruction is unpredictable.

ArchMsg.Warning.decode_unpreduseofpc

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13inreglist

Instruction is unpredictable with R13 in register-list.

ArchMsg.Warning.decode_useofd16orhigher

Attempt to use register D16 or higher where it is not allowed.

ArchMsg.Warning.decode_writebackandbaseinlist

Instruction is unpredictable with write back when the base register is in the transfer list.

ArchMsg.Warning.decode_zeroregistersinlist

Instruction is unpredictable when no registers are to be transfered.

ArchMsg.Warning.recursive_branch

An instruction is performing a branch that targets the same instruction. If this was intended then a WFI might be more effective. Fields:

PC unsigned int
: Address of instruction causing the branch

ArchMsg.Warning.recursive_exception

An instruction has generated an exception that targets the same instruction. Fields:

CPSR unsigned int
: Processor state

PC unsigned int
: Address of instruction causing the exception

TYPE enum
: Type of exception

ArchMsg.Warning.reentrant_vector_catch

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR unsigned int
vector address

ArchMsg.Warning.reserved_it_state

Fields:

ITSTATE unsigned int
reserved IT state

ArchMsg.Warning.secure_vector_fetch_from_nonsecure

Secure vector table is being fetched from Non-secure memory .

ArchMsg.Warning.unpredictable_a32_breakpoint

Fields:

ADDR1 unsigned int
Address
ADDR2 unsigned int
Breakpoint address
BAS unsigned int
BAS Field
IS_ADDRESS_MISMATCH_BREAKPOINT bool
Is Mismatch breakpoint, not a match breakpoint
N unsigned int
Breakpoint number
OFFSET unsigned int
Indicates which halfword actually considering into the A32 instruction

ArchMsg.Warning.unpredictable_banked_register_access

Fields:

R unsigned int
R bit
SYSM unsigned int
SYSm

ArchMsg.Warning.unpredictable_ccfail_undef

Fields:

OPCODE unsigned int
opcode of the instruction
VADDR unsigned int
Virtual address of the instruction being executed

ArchMsg.Warning.unpredictable_ici_bits

ICI bits do not apply or are architecturally unpredictable for instruction.

ArchMsg.Warning.unpredictable_t32_breakpoint

Fields:

ADDR1 unsigned int
Address
ADDR2 unsigned int
Breakpoint address
BAS unsigned int
BAS Field

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

IS_ADDRESS_MISMATCH_BREAKPOINT bool
: Is Mismatch breakpoint, not a match breakpoint

N unsigned int
: Breakpoint number

OFFSET unsigned int
: Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_t32_vector_catch

Fields:

ADDR unsigned int
: Address

OFFSET unsigned int
: Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_watchpoint_far

Fields:

LOWER_BOUND unsigned int
: the lowest address accessed by the instruction that triggered the watchpoint

UPPER_BOUND unsigned int
: the highest watchpointed address accessed by that instruction

ArchMsg.Warning.warning_AdvSIMDExpandImmUnexpectedZero

AdvSIMDExpandImm may treat this immediate value as UNPREDICTABLE.

ArchMsg.Warning.warning_ConditionalSMC

SMC instruction has UNPREDICTABLE effects when conditional when combined with traps.

ArchMsg.Warning.warning_access_crosses_page_boundary

Access crosses page boundary.

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault_on_both_pages

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_spanning_different_memory

Fields:

ADDR unsigned int
: address of access crossing page

MEMTYPE_PAGE1 signed int
: First Page's memory type (0-Normal, 1-Non-Normal)

MEMTYPE_PAGE2 signed int
: Second Page's memory type (0-Normal, 1-Non-Normal)

ArchMsg.Warning.warning_access_wraps_around_memory

Fields:

WIDTH unsigned int
: Address width

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_bx_from_thumbee

Fields:

ADDR unsigned int
: Destination address

ArchMsg.Warning.warning_ccsidr_unimplemented_level

Fields:

CSSELR unsigned int
: current effective value of CSSELR

ArchMsg.Warning.warning_change_to_ns_when_tge_set

Attempting to change NS to 1 when HCR.TGE = 1.

ArchMsg.Warning.warning_contiguous_bit_check_abort

Fields:

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ArchMsg.Warning.warning_contiguous_bit_error

Fields:

CONTIG_ADDR unsigned int
: address of the TLB first read, with which this entry is expected to be contiguous

CONTIG_DATA unsigned int
: expected contents based on the entry at CONTIG_ADDR

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ENTRY_DATA unsigned int
: contents of conflicting TLB entry

ArchMsg.Warning.warning_cp10_cp11_mismatch

Fields:

EL unsigned int
: EL owning the control

ArchMsg.Warning.warning_cp10_cp11_reserved_value

Fields:

CP10 unsigned int
: CP10 access

CP11 unsigned int
: CP11 access

ArchMsg.Warning.warning_csselr_level_out_of_range

Fields:

IMPLEMENTED unsigned int
: number of cache levels implemented

LEVEL unsigned int
: written value of CSSELR.Level

TYPE enum
: The type of cache selected by CSSELR.InD and TnD

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_debug_flow_control_bits_not_obeyed

Fields:

ERROR enum
: type of error resulting

REGISTER enum
: register under consideration

ArchMsg.Warning.warning_debug_register_access_during_reset

Fields:

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

ArchMsg.Warning.warning_decode_cps_inconsistent_fields

The mode setting fields in the CPS instruction are inconsistent.

ArchMsg.Warning.warning_decode_invalid_state

Instruction is unpredictable in current exception-level/security state.

ArchMsg.Warning.warning_default_cacheable_mmu_on

Default cacheable is enabled (HCR.DC = 1) while MMU is enabled.

ArchMsg.Warning.warning_default_cacheable_vmmu_off

Default cacheable is enabled (HCR.DC = 1) while HCR.VM = 0.

ArchMsg.Warning.warning_deprecated_wvr_bit_2_set

Fields:

N unsigned int
: index

WVR unsigned int
: DBGWVR

ArchMsg.Warning.warning_eret_while_software_step_active_pending

Missing ISB between setting MDSCR_EL1.SS and ERET.

ArchMsg.Warning.warning_exclusive_to_non_normal

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_exclusive_to_non_writeback

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_execute_from_device_memory

An attempt was made to execute from Device memory.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_illegal_cpsr_mode

Fields:

NEW_MODE unsigned int
: New value of CPSR.M

ArchMsg.Warning.warning_illegal_srs_mode

Fields:

MODE unsigned int
: mode for Banked SP

ArchMsg.Warning.warning_implementation_defined_read_debug_register_in_SCS

Read Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_implementation_defined_sequential_security_transitions_supported

The behaviour of a sequential instruction fetches that cross from non-secure to secure memory and contain SG instruction is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_implementation_defined_stack_limit_check_supported

It is IMPLEMENTATION DEFINED whether stack pointer limit checking is performed for the this instructions. Fields:

ADDRESS unsigned int
: Address of instruction

OPCODE unsigned int
: instruction opcode

ArchMsg.Warning.warning_implementation_defined_write_debug_register_in_SCS

Write Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

DATA unsigned int
: data attempted to be written

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_invalid_tcr_granule

Fields:

REQUEST signed int
: page size requested (or 0 for reserved)

SUBSTITUTE signed int
: best guess available page size

TG_ID bool
: bits TG0 or TG1

ArchMsg.Warning.warning_load_multiple_user_registers_from_user_mode

An LDM(user registers) instruction executed from user mode.

ArchMsg.Warning.warning_load_pc_from_unaligned

PC loaded from an unaligned location.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_mte_in_mem_attr_not_supported

Fields:

ATTR unsigned int
: Memory attribute

ArchMsg.Warning.warning_reserved_breakpoint_state_match

Fields:

IS_BREAKPOINT unsigned int
: Is breakpoint, not watchpoint

ArchMsg.Warning.warning_shareability

Unpredictable: combination of the 1st and 2nd stages of translation is Normal Inner Non-Cacheable, Outer Non-Cacheable.

ArchMsg.Warning.warning_software_step_set_while_enabled

MDSCR_EL1.SS set to 1 while software step debug exceptions are enabled.

ArchMsg.Warning.warning_thumb_instruction_wraps_around_memory

Thumb instruction warps around memory.

ArchMsg.Warning.warning_ttbr_sbz_bits_are_not_zero

Fields:

MSB unsigned int
: x-1

TTBR enum
: which TTBR register

ArchMsg.Warning.warning_unaligned_address_dbgdtrrx_write

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_unaligned_address_dbgdtrtx_read

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_unaligned_to_device

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_unaligned_to_strongly_ordered

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_unknown_sau_rnr

SAU_RNR was set to an unsupported value.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_AIRCR_PRIP_and_BFHFNMINP

The effect of setting both AIRCR.BFHFNMINP and AIRCR.PRIP to 1 is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTCLRACTIVE_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTCLRACTIVE if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_and_SYSRESETREQ

When the processor is halted in Debug state, if a write to the register writes a 1 to both VECTRESET and SYSRESETREQ, the behavior is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTRESET if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_incorrect_VKEY

The value 0x05FA must be written to AIRCR.VKEY, otherwise the register write is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_EXE_RETURN_Reserved_Bit

EXC_RETURN[23:7] are reserved with the special condition that all bits should be written as one. Values other than all 1s are UNPREDICTABLE. Fields:

EXC_RETURN `unsigned int`
: Special PC value

SBO_MASK `unsigned int`
: Which bits Should Be One to not UNPRED

ArchMsg.Warning.warning_unpredictable_change_to_DEMCR_MON_STEP_at_insufficient_priority

The effect of changing DEMCR.MON_STEP at an execution priority that is lower than the priority of the DebugMonitor exception is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_change_to_priority_of_active_exception

Changing the priority of an active exception is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_clear_lspact

Fields:

DRVEXC_TAKEN `unsigned int`
: derived exception taken not pended

REACHED_LAST_STORE `unsigned int`
: only the last faulted

ArchMsg.Warning.warning_unpredictable_dbg_exit_unaligned_dlr

Debug exit to AArch32 with DLR[0] set to 1. PC[0] can be set to 0 or to DLR[0].

ArchMsg.Warning.warning_unpredictable_exception_catch

Generation of an exception catch event is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_inconsistent_state

state on exception return is unpredictable.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_exception_return_instruction

exception return instruction is unpredictable.

ArchMsg.Warning.warning_unpredictable_in_debug_state

Instruction is UNPREDICTABLE when executed in debug state.

ArchMsg.Warning.warning_unpredictable_mair_encoding

Fields:

ATTR unsigned int
: Memory attribute

ArchMsg.Warning.warning_unpredictable_pmu_counter_access

Access to PMU counters from non-secure EL0 or EL1 is UNPREDICTABLE with MDCR_EL2.HPMN set to 0.

ArchMsg.Warning.warning_unpredictable_prioritization_breakpoint_match_vector_catch

Two events with the same priority occurred at the same instruction: Address Matching Vector Catch debug event, and Breakpoint debug event It is constrained unpredictable which is taken.

ArchMsg.Warning.warning_unpredictable_stack_selection

UNPREDICTABLE state on context switch.

ArchMsg.Warning.warning_unpredictable_tsize_out_of_range

Fields:

TCR enum
: Which TCR

TSIZE signed int
: Value of TSize

TSIZE_MAX signed int
: The maximum allowed value

TTBR unsigned int
: Which TTBR region

ArchMsg.Warning.warning_unpredictable_unaligned_pc_as_base_register

The PC value must be word-aligned, otherwise the behavior of the instruction is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_unaligned_pop_stack

Fields:

FRAME_ADDR unsigned int
: unaligned stack frame address

ArchMsg.Warning.warning_unpredictable_vmsa_memattrib

Fields:

TEXCB unsigned int
: texcb value

TEX_REMAP bool
: use tex remap

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_vtcr_t0sz_sl0

Fields:

SL0 signed int
: VTCR.SL0, starting level for stage 2 translation table walks

T0SZ signed int
: VTCR.T0SZ, required input address range

ArchMsg.Warning.warning_unpredictable_write_DHCSR

Fields:

NEW unsigned int
: data attempted to be written

OLD unsigned int
: previous value

ArchMsg.Warning.warning_unpredictable_write_SHCSR

UNPREDICTABLE setting a pending bit to 1 for an exception with priority >= execution priority.

ArchMsg.Warning.warning_unsupported_access_to_memory_mapped_register

Access to memory mapped register is unsupported.

ArchMsg.Warning.warning_user_jmcr_access

Fields:

WRITE unsigned int
: access is write

ArchMsg.Warning.warning_wcr_mask_and_bas

DBGWCR_EL1.MASK is non-zero and BAS is not 0xff.

ArchMsg.Warning.warning_wcr_mask_reserved

DBGWCR_ELn.MASK is set to a reserved value.

ArchMsg.Warning.warning_wcr_non_configuous_bas

DBGWCR_EL1.BAS has non-contiguous set of ones.

ArchMsg.Warning.warning_wvr_masked_bit_not_zero

DBGWVRn_EL1.VA contains a bit masked by DBGWCRn_EL1.MASK that is not zero.

BRANCH_MISPREDICT

Simulating branch mispredict. Fields:

PC unsigned int
: Origin address (or 0 if unavailable)

BRA_DIR

Direct branches, to immediate address. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR

Indirect branches, perhaps to a register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR_CCFAIL

Indirect branches fail, perhaps to a register.

BRA_RET_CCFAIL

Branche return fail, perhaps to a register.

CACHE_MAINTENANCE_OP

Cache Maintenance Operation. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Specified MVA or Set/way

FUNCTION enum
: Clean / Invalidate

SCOPE enum
: Affected region

SIDE enum
: Inst / Data.

CCFAIL

Conditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCFAIL_UNC

Unconditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS

Conditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS_UNC

Unconditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CODE_CACHE_MAINT

Code cache maintenance. Fields:

END_INCL unsigned int
: Inclusive end addres for by VA or PA requests

NS bool
: Non-secure world for PA

START unsigned int
: Start address for by VA or PA requests

TYPE enum
: Request type.

COMPILE_BLOCK_END

Last instruction of basic block translated. Fields:

VADDR unsigned int
: Address of next instruction after this basic block

COMPILE_BLOCK_START

First instruction of basic block translated. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VADDR unsigned int
: Address of first instruction in basic block

COMPILE_INST

ARM instruction compiled. Fields:

DISASS string
: Disassembly of the instruction.

ISET enum
: The instruction set of this instruction.

ITSTATE unsigned int
: The ITSTATE current for the instruction.

OPCODE unsigned int
: The opcode of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

CONTEXTIDR

Context ID Register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

NS bool
: Secure or nonsecure banked register is accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

CONTEXT_SYNC

Called for every context synchronization event. Fields:

ARCH_CSE bool
: Architectural Context Synchronization Event

SYSREG_SYNC bool
: System register synchronization event

CORE_ENDIAN

Core BE8 Big-Endian state changed. Fields:

BE8 bool
: Core BE8 Big-Endian state

CORE_INFO

Static processor attributes. Only triggered by a call to DumpState(). Fields:

ARCH_PROFILE enum
: The architecture profile of the core.

CLUSTER_ID unsigned int
: The cluster ID of this processor.

CORE_NUM unsigned int
: The number of this core in an MP processor.

DETAILED_CORE_FEATURES enum
: Detailed core features as a bitfield. Enum values are bitmasks to extract the info from the field

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

FPU_VERSION enum
: The VFP version implemented by the core.

MEM_ARCH enum
: The memory architecture of the core.

MIDR_EL1 unsigned int
: MIDR_EL1 value

NUM_CORES unsigned int
: The number of cores in this MP processor.

QUANTUM_SIZE unsigned int
: The default quantum size of the core.

SECURITY_FEATURES bool
: Does the core have security features?

CORE_LOADS

Processor load accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_REGS

Changes of the core registers R0 to R14. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number, 0 to 14.

MODE enum
: Bank of the register accessed.

OLD_VALUE unsigned int
: The old value overwritten.

PHYS_ID enum
: The physical register accessed.

VALUE unsigned int
: The new value written to the register.

CORE_STORES

Processor store accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CP14_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP14_WRITE

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

CP15_WRITE

System Control Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

opc2 unsigned int
: opcode 2

CP15_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

CPSR

CPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

OLD_VALUE unsigned int
: The old CPSR value

UNKNOWN unsigned int
: Bits within the register that have unknown value.

VALUE unsigned int
: The new CPSR value

CRYPTO_SPEC

Every crypto instruction speculatively executed.

CorePowerStateChange

Core power-up state has changed. Fields:

PU bool
: Whether the core powered up

DATA_CACHE_SET_ALLOC_TAGS

Set allocation tags from DC GVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of memory in bytes.

TAG unsigned int
: The value of the tag being set.

VADDR unsigned int
: Virtual address used for setting allocation tags.

DATA_CACHE_ZERO

Zero and invalidate cache line from DC ZVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of zero'd memory in bytes.

VADDR unsigned int
: Virtual address of the zero'd cache line.

DEBUG_EVENT

Hardware debug support event. Fields:

EVENT enum
: Description of event

VALUE unsigned int
: data value

DTCM_ACCESS

Trace TCM accesses.

DTCM_ACCESS_RAZ_WI

Trace TCM accesses. Fields:

ADDR unsigned int
: Address of the RAZ/WI access.

END_COMPILE

Compilation end. Fields:

END_OF_PAGE_COUNT unsigned int
: Number of basic blocks exited due to page boundary since START_COMPILE.

FETCHFAIL_COUNT unsigned int
: Number of basic blocks exited due to ifetch failure START_COMPILE.

INST_COUNT unsigned int
: Number of instructions compiled since START_COMPILE.

NONSEQ_COUNT unsigned int
: Number of basic blocks exited due to non-sequential instructions since START_COMPILE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PAGE_STRADDLE_COUNT unsigned int
: Number of basic blocks exited due to unaligned instructions crossing page since START_COMPILE.

EXCEPTION

Exception taken. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

IS_PHYSICAL bool
: Physical or Virtual exception

LR unsigned int
: The value assigned to the link register.

PC unsigned int
: The location where the exception occurred.

PREFERRED_RETURN unsigned int
: The preferred return address for the exception.

TARGET_ISET enum
: The instruction set of the exception handler code.

TARGET_PC unsigned int
: The address the exception branches to.

VECTOR enum
: The exception vector.

VECTOR_OFFSET unsigned int
: The offset of an exception vector from the base address.

EXCEPTION_END

Every exception completed.

EXCEPTION_RAISE

Every exception raised.

EXCEPTION_RETURN

Branches on leaving exception. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

EXCEPTION_RETURN_PREBRANCH

Exception return event that must occur before PC and processor state have updated.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

EXCEPTION_START

Every exception started.

FP_STATE

Floating point state. Fields:

FPSR `unsigned int`
: FPSR register value.

MASK `unsigned int`
: The mask indicating updated cumulative bits.

HLT

HLT instruction occurred. Fields:

IMM16 `unsigned int`
: The 16-bit immediate value encoded in HLT instruction.

INST

Every instruction executed. Fields:

CORE_NUM `unsigned int`
: Core number in a multi processor.

CURRENT_TIME `unsigned int`
: The core's current time, as simulated time plus local time.

DEBUG_STATE `bool`
: The instruction is executed in debug state.

DISASS `string`
: Disassembly of instruction.

INST_COUNT `unsigned int`
: The core's instruction counter, starting at 1 for the first instruction.

ISET `enum`
: The current instruction set.

ITSTATE `unsigned int`
: The current ITSTATE.

LOCAL_TIME `unsigned int`
: The core's local time, relative to the current quantum.

MODE `enum`
: The mode the core is in.

NSDESC `unsigned int`
: The physical address non-secure bit.

NSDESC2 `unsigned int`
: The second page physical address non-secure bit.

OPCODE `unsigned int`
: The opcode of the instruction.

PADDR `unsigned int`
: The physical address of the instruction.

PADDR2 `unsigned int`
: If different from PADDR, the physical address of the second page of the instruction.

PC `unsigned int`
: The address of the instruction.

SIZE `unsigned int`
: The size of the instruction in bytes.

INST_END

Every instruction completed.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_START

Every instruction started. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

INST_STRADDLE

Instruction straddles boundary. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

ISB

ISB instruction occurred.

ITCM_ACCESS

Trace TCM accesses.

ITCM_ACCESS_RAZ_WI

Trace TCM accesses. Fields:

ADDR unsigned int
: Address of the RAZ/WI access.

LOAD_MULTIPLE_REGS_LIST

List of destination registers (not source) of a multiple registers load instruction. Fields:

REG string
: Register to add into the list

LOCAL_MONITOR

Local monitor activity. Fields:

PADDR unsigned int
: Local Monitor Address

State enum
: State of the monitor (Open/Exclusive)

MEMMAP_DEBUG_READ

Memory mapped reads to the debug registers. Fields:

ADDR unsigned int
: address

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

MEMMAP_DEBUG_WRITE

Memory mapped writes to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

MODE_CHANGE

Mode change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: The new mode.

OLD_MODE enum
: The old mode.

MPU_TRANS

Address translation information. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

EXEC_PERM enum
: Execution Permission

INNERCACHE_RA bool
: Is the inner cache allocate on read

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

MEMTYPE enum
: Memory type.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

PADDR unsigned int
: Address of the access.

PAGESIZE unsigned int
: Page size as log2(size).

READ_PERM enum
: Read Permission

REG_NUM signed int
: MPU region-number 0-15 (else 0xFF=none, 0xBB/0xDD=background/default)

SH enum
: Shareability

SIDE enum
: Inst / Data.

WRITE_PERM enum
: Write Permission

PERIODIC

Called for every quantum. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

PC unsigned int
: The address of the next instruction to be executed on this CPU.

PMU_COUNTER_OVERFLOW

PMU counter overflow. Fields:

EVENT_ID unsigned int
: Event Identifier

INDEX unsigned int
: Counter index (as selected by PMSELR).

INTERRUPT bool
: Is interrupt enabled on overflow for this counter

PRELOAD_DATA

Data preload from PLD instruction. Fields:

VADDR unsigned int
: Virtual address of the data that should be preloaded.

PRELOAD_INST

Instruction preload from PLI instruction. Fields:

VADDR unsigned int
: Virtual address of the instruction that should be preloaded.

PRE_CORE_LOAD

Trace just before a core load. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_CORE_STORE

Trace just before a core store. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PSTATE_PACTIVE_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

PSTATE_PACTIVE_OPMODE_TRANSITION_DENIED

Trace the denied pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_POWER_TRANSITION

Trace pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

PSTATE_PACTIVE_POWER_TRANSITION_DENIED

Trace the denied pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

REASON string
: Reason for the transition being denied

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PSTATE_PACTIVE_THREAD_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions for threads on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

RUN_STATE

Run state transition. Fields:

INST_COUNT unsigned int
: Ticks count at point of transition.

NEW enum
: New run state.

OLD enum
: Old run state.

SEMIHOSTING_CALL

Call of a semihost function occurred. Fields:

PC unsigned int
: The program counter after the semihosting call.

SEMIHOSTING_PRECALL

About to call semihost. Fields:

EL2 bool
: Use EL2 translation regime.

NS bool
: Is Non-Secure.

REG_WIDTH unsigned int
: The current register width in bytes.

SIGNAL

External signal state change. Fields:

SIGNAL enum
: Signal that changed

STATE bool
: Signal asserted state

SPSR

SPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: Which of the banked SPSR registers is written.

OLD_VALUE unsigned int
: The old SPSR value

VALUE unsigned int
: The new SPSR value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

START_COMPILE

Compilation started. Fields:

VADDR unsigned int
: Instruction where compilation begins.

STG_MULTIPLE_MICRO_OPS_ADDRESS_RANGE

Address range for MTE stores with multiple uops. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

STORE_MULTIPLE_REGS_ADDRESS_RANGE

Address range for stores with multiple registers. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

SUBOPTIMAL_LDST_RETIRED

suboptimal load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1: half-word, 2: word ...)

OPTIMAL_ALIGN_SELECTOR unsigned int
: Selector to decide the optimal alignment size (1: access width, 2: 4-bytes, 3: 8-bytes, ..., 12: 4-kbytes)

VADDR unsigned int
: The virtual address of the access

SYNC

Called for every synchronization, that is every quantum break. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

LOCAL_QUANTUM unsigned int
: The local quantum of this CPU.

LOCAL_TIME unsigned int
: The local time of this CPU.

SYNC_REASON enum
: Reason for sync.

SYSCALL

System call instruction executed. Fields:

IMM unsigned int
: Immediate value of the system call instruction.

TYPE enum
: System call type.

VADDR unsigned int
: Instruction that caused the system call.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SYSREG_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE32

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE64

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

UNALIGNED_LDST_RETIRED

Processor unaligned load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1=>half-word, 2->word ...)

NUMBER_OF_BEATS unsigned int
: Number of accesses (beats) in this burst (Total size in bytes is ACCESS_SIZE * NUMBER_OF_BEATS)

VADDR unsigned int
: The virtual address of the access

VFP_D_REGS

VFP/NEON D 64 bit register write. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_Q_REGS

VFP/NEON Q 128 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding byte in the register. For example, 0x1 would indicate a write to VALUE[7:0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_SYS_REGS

Writes to the VFP/NEON units system registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: Which VFP system register is written.

MASK unsigned int
: The mask indicating updated cumulative bits.

OLD_VALUE unsigned int
: The register's old value overwritten.

VALUE unsigned int
: The new value written to the VFP system register.

VFP_S_REGS

VFP/NEON S 32 bit register write. Fields:

ALIASING enum
: Type of register aliasing used.

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

WAYPOINT

Signals end of basic block execution. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: Origin instructions set

IS_COND bool
: Indicates if this is a conditional waypoint.

PC unsigned int
: Origin address (or 0 if unavailable)

TAKEN bool
: Indicates if this waypoint was taken.

TARGET unsigned int
: Destination address

TARGET_ISET enum
: Destination instructions set

WFE_END

WFE ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFE.

WFE_EVENT_REGISTER

WFE event register status: set/clear, reason. Fields:

INST_COUNT unsigned int
: Ticks count.

REASON enum
: Reason for set/clear. Only REASON==0 (Cleared by WFE) clears the bit, all other reasons set it.

WFE_IGNORED

WFE ignored. Fields:

EVENT bool
: This WFE was ignored because the event register was set.

INST_COUNT unsigned int
: Ticks count when ignoring WFE.

TRAPPED bool
: This WFE was trapped.

WFE_START

WFE entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

WFI_END

WFI ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFI.

WFI_IGNORED

WFI ignored. Fields:

DISABLED bool
: This WFI was ignored because WFI is disabled.

INST_COUNT unsigned int
: Ticks count when ignoring WFI.

REASON enum
: specifies reason why WFI trace was ignored

TRAPPED bool
: This WFI was trapped.

WFI_START

WFI entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFI.

WFI_WAKEUP

WFI wakeup. Fields:

INST_COUNT unsigned int
: Ticks count when WFI wakeup occurred.

REASON enum
: Reason for wakeup.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.42 ARM_Cortex-X1 - trace

This section describes the trace sources.

ASYNC_MEMORY_FAULT

Context ID Register write. Fields:

FAULT unsigned int
: Fault status in ESR format

PADDR unsigned int
: Physical Address (or 0 if unavailable)

VADDR signed int
: Virtual Address (or 0 if unavailable)

ATOMIC_END_ACCESS

Bus trace access for atomics. Fields:

ACCESS_FAIL bool
: Memory access failed

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

LOAD_VALUE unsigned int
: Loaded values

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ATOMIC_START_ACCESS

Bus trace access for atomics. Fields:

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

NSDESC unsigned int
: The security state of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ArchMsg.Error.exit_code

Fields:

EXITCODE signed int
: Exit code to be returned

KIND string
: Component kind that invoked the exit code trace

REASON string
: Reason why the exit code trace was invoked (optional)

ArchMsg.Warning.IgnoredWarmResetRequest

Event fires when software attempts to request warm reset through write to deprecated DBGPRCR.CWRR.

ArchMsg.Warning.RAS_ErrselrUnpredictableState

ERRSELR is in unpredictable state due to value greater than number_of_error_records written to it.

ArchMsg.Warning.RAS_UnpredictableStateWrite

Fields:

REG_NAME string
: Name of register being written.

ArchMsg.Warning.TarmacReconstructor

Unexpected event happened during Tarmac reconstruction. Fields:

REASON string
: Reason

ArchMsg.Warning.Unpredictably Indexed PM Event Register

Fields:

IsDirect unsigned int
: Direct Access rather than indirect

N unsigned int
: Number of Accessible Registers

SEL unsigned int
: Selector

ArchMsg.Warning.branch_to_unaligned_address

Fields:

ADDR unsigned int
: Unaligned branch address

ArchMsg.Warning.cache_contents_unknown

A cache has been activated without being invalidated and may contain UNKNOWN content. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

REGIME unsigned int
: Current translation regime

SIDE unsigned int
: 0 d-side cache or unified cache, 1 i-side

ArchMsg.Warning.dcimvac_matches_watchpoint

Watchpoints matching an AArch32 DCIMVA are implementation defined.

ArchMsg.Warning.decode_fieldmismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_fracbitsnegative

VCVT Instruction is unpredictable with a negative number of fraction bits.

ArchMsg.Warning.decode_initblock

Instruction is unpredictable in an IT block.

ArchMsg.Warning.decode_initblocknotlast

Instruction is unpredictable in an IT block when not the last.

ArchMsg.Warning.decode_invalid_condition

Instruction is unpredictable when the condition is not AL.

ArchMsg.Warning.decode_invalidfieldcombination

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_invalidvalue

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_mem_hint_unallocated

There is no memory hint allocated to this bit pattern.

ArchMsg.Warning.decode_registermatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_registermismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registeroutofrange

Float point instruction is unpredictable indexing beyond the end of the register bank.

ArchMsg.Warning.decode_sbzsbo

Reserved bits in the instruction are not canonical.

ArchMsg.Warning.decode_transactiontoobig

Attempt to perform a memory transaction of more than 128 bytes.

ArchMsg.Warning.decode_unpred_other

Instruction is unpredictable.

ArchMsg.Warning.decode_unpreduseofpc

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13inreglist

Instruction is unpredictable with R13 in register-list.

ArchMsg.Warning.decode_useofd16orhigher

Attempt to use register D16 or higher where it is not allowed.

ArchMsg.Warning.decode_writebackandbaseinlist

Instruction is unpredictable with write back when the base register is in the transfer list.

ArchMsg.Warning.decode_zeroregistersinlist

Instruction is unpredictable when no registers are to be transfered.

ArchMsg.Warning.recursive_branch

An instruction is performing a branch that targets the same instruction. If this was intended then a WFI might be more effective. Fields:

PC unsigned int
: Address of instruction causing the branch

ArchMsg.Warning.recursive_exception

An instruction has generated an exception that targets the same instruction. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CPSR unsigned int
: Processor state

PC unsigned int
: Address of instruction causing the exception

TYPE enum
: Type of exception

ArchMsg.Warning.reentrant_vector_catch

Fields:

ADDR unsigned int
: vector address

ArchMsg.Warning.reserved_it_state

Fields:

ITSTATE unsigned int
: reserved IT state

ArchMsg.Warning.tlb_contents_unknown

Translation has been enabled without invalidating the TLB. Fields:

EL unsigned int
: Exception level of translation regime

INVALIDITY unsigned int
: Which TLB may be invalid (if TLBs are separate)

NON_SECURE unsigned int
: Security state of translation regime

VMID unsigned int
: Current VMID, if applicable

ArchMsg.Warning.unknown_DLR_DSPSR

Exiting debug state with unknown DLR or DSPSR.

ArchMsg.Warning.unknown_ELR_SPSR

Returning from debug exception with unknown ELR or SPSR.

ArchMsg.Warning.unpredictable_a32_breakpoint

Fields:

ADDR1 unsigned int
: Address

ADDR2 unsigned int
: Breakpoint address

BAS unsigned int
: BAS Field

IS_ADDRESS_MISMATCH_BREAKPOINT bool
: Is Mismatch breakpoint, not a match breakpoint

N unsigned int
: Breakpoint number

OFFSET unsigned int
: Indicates which halfword actually considering into the A32 instruction

ArchMsg.Warning.unpredictable_banked_register_access

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

R unsigned int
R bit
SYSM unsigned int
SYSm

ArchMsg.Warning.unpredictable_ccfail_undef

Fields:

OPCODE unsigned int
opcode of the instruction
VADDR unsigned int
Virtual address of the instruction being executed

ArchMsg.Warning.unpredictable_ici_bits

ICI bits do not apply or are architecturally unpredictable for instruction.

ArchMsg.Warning.unpredictable_t32_breakpoint

Fields:

ADDR1 unsigned int
Address
ADDR2 unsigned int
Breakpoint address
BAS unsigned int
BAS Field
IS_ADDRESS_MISMATCH_BREAKPOINT bool
Is Mismatch breakpoint, not a match breakpoint
N unsigned int
Breakpoint number
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_t32_vector_catch

Fields:

ADDR unsigned int
Address
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_watchpoint_far

Fields:

LOWER_BOUND unsigned int
the lowest address accessed by the instruction that triggered the watchpoint
UPPER_BOUND unsigned int
the highest watchpointed address accessed by that instruction

ArchMsg.Warning.warning_AdvSIMDExpandImmUnexpectedZero

AdvSIMDExpandImm may treat this immediate value as UNPREDICTABLE.

ArchMsg.Warning.warning_ConditionalSMC

SMC instruction has UNPREDICTABLE effects when conditional when combined with traps.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_access_crosses_page_boundary

Access crosses page boundary.

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault_on_both_pages

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_spanning_different_memory

Fields:

ADDR unsigned int
: address of access crossing page

MEMTYPE_PAGE1 signed int
: First Page's memory type (0-Normal, 1-Non-Normal)

MEMTYPE_PAGE2 signed int
: Second Page's memory type (0-Normal, 1-Non-Normal)

ArchMsg.Warning.warning_access_wraps_around_memory

Fields:

WIDTH unsigned int
: Address width

ArchMsg.Warning.warning_bx_from_thumbee

Fields:

ADDR unsigned int
: Destination address

ArchMsg.Warning.warning_ccsidr_unimplemented_level

Fields:

CSSELR unsigned int
: current effective value of CSSELR

ArchMsg.Warning.warning_change_to_ns_when_tge_set

Attempting to change NS to 1 when HCR.TGE = 1.

ArchMsg.Warning.warning_contiguous_bit_check_abort

Fields:

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ArchMsg.Warning.warning_contiguous_bit_error

Fields:

CONTIG_ADDR unsigned int
: address of the TLB first read, with which this entry is expected to be contiguous

CONTIG_DATA unsigned int
: expected contents based on the entry at CONTIG_ADDR

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ENTRY_ADDR unsigned int
address of conflicting TLB entry
ENTRY_DATA unsigned int
contents of conflicting TLB entry

ArchMsg.Warning.warning_cp10_cp11_mismatch

Fields:

EL unsigned int
EL owning the control

ArchMsg.Warning.warning_cp10_cp11_reserved_value

Fields:

CP10 unsigned int
CP10 access
CP11 unsigned int
CP11 access

ArchMsg.Warning.warning_csselr_level_out_of_range

Fields:

IMPLEMENTED unsigned int
number of cache levels implemented
LEVEL unsigned int
written value of CSSELR.Level
TYPE enum
The type of cache selected by CSSELR.InD and TnD

ArchMsg.Warning.warning_debug_flow_control_bits_not_obeyed

Fields:

ERROR enum
type of error resulting
REGISTER enum
register under consideration

ArchMsg.Warning.warning_debug_register_access_during_reset

Fields:

IS_WRITE unsigned int
Write Not Read
OFFSET unsigned int
Register Offset

ArchMsg.Warning.warning_decode_cps_inconsistent_fields

The mode setting fields in the CPS instruction are inconsistent.

ArchMsg.Warning.warning_decode_invalid_state

Instruction is unpredictable in current exception-level/security state.

ArchMsg.Warning.warning_default_cacheable_mmu_on

Default cacheable is enabled (HCR.DC = 1) while MMU is enabled.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_default_cacheable_vmmu_off

Default cacheable is enabled (HCR.DC = 1) while HCR.VM = 0.

ArchMsg.Warning.warning_deprecated_wvr_bit_2_set

Fields:

N unsigned int
: index

WVR unsigned int
: DBGWVR

ArchMsg.Warning.warning_eret_while_software_step_active_pending

Missing ISB between setting MDSCR_EL1.SS and ERET.

ArchMsg.Warning.warning_exclusive_to_non_normal

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_exclusive_to_non_writeback

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_execute_from_device_memory

An attempt was made to execute from Device memory.

ArchMsg.Warning.warning_illegal_cpsr_mode

Fields:

NEW_MODE unsigned int
: New value of CPSR.M

ArchMsg.Warning.warning_illegal_srs_mode

Fields:

MODE unsigned int
: mode for Banked SP

ArchMsg.Warning.warning_implementation_defined_read_debug_register_in_SCS

Read Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_implementation_defined_sequential_security_transitions_supported

The behaviour of a sequential instruction fetches that cross from non-secure to secure memory and contain SG instruction is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_implementation_defined_stack_limit_check_supported

It is IMPLEMENTATION DEFINED whether stack pointer limit checking is performed for the this instructions. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDRESS unsigned int
: Address of instruction

OPCODE unsigned int
: instruction opcode

ArchMsg.Warning.warning_implementation_defined_write_debug_register_in_SCS

Write Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

DATA unsigned int
: data attempted to be written

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_invalid_tcr_granule

Fields:

REQUEST signed int
: page size requested (or 0 for reserved)

SUBSTITUTE signed int
: best guess available page size

TG_ID bool
: bits TG0 or TG1

ArchMsg.Warning.warning_load_multiple_user_registers_from_user_mode

An LDM(user registers) instruction executed from user mode.

ArchMsg.Warning.warning_load_pc_from_unaligned

PC loaded from an unaligned location.

ArchMsg.Warning.warning_mte_in_mem_attr_not_supported

Fields:

ATTR unsigned int
: Memory attribute

ArchMsg.Warning.warning_reserved_breakpoint_state_match

Fields:

IS_BREAKPOINT unsigned int
: Is breakpoint, not watchpoint

ArchMsg.Warning.warning_shareability

Unpredictable: combination of the 1st and 2nd stages of translation is Normal Inner Non-Cacheable, Outer Non-Cacheable.

ArchMsg.Warning.warning_software_step_set_while_enabled

MDSCR_EL1.SS set to 1 while software step debug exceptions are enabled.

ArchMsg.Warning.warning_thumb_instruction_wraps_around_memory

Thumb instruction warps around memory.

ArchMsg.Warning.warning_ttbr_sbz_bits_are_not_zero

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MSB unsigned int
x-1
TTBR enum
which TTBR register

ArchMsg.Warning.warning_unaligned_address_dbgdtrrx_write

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_address_dbgdtrtx_read

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_device

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_strongly_ordered

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unknown_sau_rnr

SAU_RNR was set to an unsupported value.

ArchMsg.Warning.warning_unpredictable_AIRCR_PRIP_and_BFHFNMINP

The effect of setting both AIRCR.BFHFNMINP and AIRCR.PRIP to 1 is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTCLRACTIVE_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTCLRACTIVE if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_and_SYSRESETREQ

When the processor is halted in Debug state, if a write to the register writes a 1 to both VECTRESET and SYSRESETREQ, the behavior is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTRESET if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_incorrect_VKEY

The value 0x05FA must be written to AIRCR.VKEY, otherwise the register write is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_EXE_RETURN_Reserved_Bit

EXC_RETURN[23:7] are reserved with the special condition that all bits should be written as one. Values other than all 1s are UNPREDICTABLE. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

EXC_RETURN unsigned int
Special PC value
SBO_MASK unsigned int
Which bits Should Be One to not UNPRED

ArchMsg.Warning.warning_unpredictable_change_to_DEMCR_MON_STEP_at_insufficient_priority

The effect of changing DEMCR.MON_STEP at an execution priority that is lower than the priority of the DebugMonitor exception is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_change_to_priority_of_active_exception

Changing the priority of an active exception is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_clear_lspact

Fields:

DRVEXC_TAKEN unsigned int
derived exception taken not pended
REACHED_LAST_STORE unsigned int
only the last faulted

ArchMsg.Warning.warning_unpredictable_dbg_exit_unaligned_dlr

Debug exit to AArch32 with DLR[0] set to 1. PC[0] can be set to 0 or to DLR[0].

ArchMsg.Warning.warning_unpredictable_exception_catch

Generation of an exception catch event is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_inconsistent_state

state on exception return is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_instruction

exception return instruction is unpredictable.

ArchMsg.Warning.warning_unpredictable_in_debug_state

Instruction is UNPREDICTABLE when executed in debug state.

ArchMsg.Warning.warning_unpredictable_mair_encoding

Fields:

ATTR unsigned int
Memory attribute

ArchMsg.Warning.warning_unpredictable_pmu_counter_access

Access to PMU counters from non-secure EL0 or EL1 is UNPREDICTABLE with MDCR_EL2.HPMN set to 0.

ArchMsg.Warning.warning_unpredictable_prioritization_breakpoint_match_vector_catch

Two events with the same priority occurred at the same instruction: Address Matching Vector Catch debug event, and Breakpoint debug event It is constrained unpredictable which is taken.

ArchMsg.Warning.warning_unpredictable_stack_selection

UNPREDICTABLE state on context switch.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_tsize_out_of_range

Fields:

TCR enum
: Which TCR

TSIZE signed int
: Value of TSize

TSIZE_MAX signed int
: The maximum allowed value

TTBR unsigned int
: Which TTBR region

ArchMsg.Warning.warning_unpredictable_unaligned_pc_as_base_register

The PC value must be word-aligned, otherwise the behavior of the instruction is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_unaligned_pop_stack

Fields:

FRAME_ADDR unsigned int
: unaligned stack frame address

ArchMsg.Warning.warning_unpredictable_vmsa_memattrib

Fields:

TEXCB unsigned int
: texcb value

TEX_REMAP bool
: use tex remap

ArchMsg.Warning.warning_unpredictable_vtcr_t0sz_sl0

Fields:

SL0 signed int
: VTCR.SL0, starting level for stage 2 translation table walks

T0SZ signed int
: VTCR.T0SZ, required input address range

ArchMsg.Warning.warning_unpredictable_write_DHCSR

Fields:

NEW unsigned int
: data attempted to be written

OLD unsigned int
: previous value

ArchMsg.Warning.warning_unpredictable_write_SHCSR

UNPREDICTABLE setting a pending bit to 1 for an exception with priority >= execution priority.

ArchMsg.Warning.warning_unsupported_access_to_memory_mapped_register

Access to memory mapped register is unsupported.

ArchMsg.Warning.warning_user_jmcr_access

Fields:

WRITE unsigned int
: access is write

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_wcr_mask_and_bas

DBGWCR_EL1.MASK is non-zero and BAS is not 0xff.

ArchMsg.Warning.warning_wcr_mask_reserved

DBGWCR_ELn.MASK is set to a reserved value.

ArchMsg.Warning.warning_wcr_non_configuous_bas

DBGWCR_EL1.BAS has non-contiguous set of ones.

ArchMsg.Warning.warning_wvr_masked_bit_not_zero

DBGWVRn_EL1.VA contains a bit masked by DBGWCRn_EL1.MASK that is not zero.

ArchMsg.Why.why_branch_target_exception

A branch target exception has occurred. Fields:

BTYPE unsigned int
: pstate.BTYPE at time branch taken

OPCODE unsigned int
: opcode at branch target

PC unsigned int
: Virtual Address of branch target

ArchMsg.Why.why_illegal_state

An illegal mode change has occured. Fields:

MESSAGE string
: The message

BRANCH_MISPREDICT

Simulating branch mispredict. Fields:

PC unsigned int
: Origin address (or 0 if unavailable)

BRA_DIR

Direct branches, to immediate address. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

BRA_INDIR

Indirect branches, perhaps to a register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR_CCFAIL

Indirect branches fail, perhaps to a register.

BRA_RET_CCFAIL

Branche return fail, perhaps to a register.

CACHE_MAINTENANCE_OP

Cache Maintenance Operation. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Specified MVA or Set/way

FUNCTION enum
: Clean / Invalidate

SCOPE enum
: Affected region

SIDE enum
: Inst / Data.

CCFAIL

Conditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCFAIL_UNC

Unconditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS

Conditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS_UNC

Unconditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CHECKPOINT_MESSAGE

Report error messages from the checkpointing process. Fields:

message string
: Message contents.

CHECKPOINT_RESTORE_END

Checkpoint restore completed.

CHECKPOINT_RESTORE_START

Checkpoint restore about to start.

CHECKPOINT_SAVE_END

Checkpoint save completed.

CHECKPOINT_SAVE_START

Checkpoint save about to start.

CODE_CACHE_MAINT

Code cache maintenance. Fields:

END_INCL unsigned int
: Inclusive end addres for by VA or PA requests

NS bool
: Non-secure world for PA

START unsigned int
: Start address for by VA or PA requests

TYPE enum
: Request type.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

COMPILE_BLOCK_END

Last instruction of basic block translated. Fields:

VADDR unsigned int
: Address of next instruction after this basic block

COMPILE_BLOCK_START

First instruction of basic block translated. Fields:

VADDR unsigned int
: Address of first instruction in basic block

COMPILE_INST

ARM instruction compiled. Fields:

DISASS string
: Disassembly of the instruction.

ISET enum
: The instruction set of this instruction.

ITSTATE unsigned int
: The ITSTATE current for the instruction.

OPCODE unsigned int
: The opcode of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

CONTEXTIDR

Context ID Register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

NS bool
: Secure or nonsecure banked register is accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

CONTEXT_SYNC

Called for every context synchronization event. Fields:

ARCH_CSE bool
: Architectural Context Synchronization Event

SYSREG_SYNC bool
: System register synchronization event

CORE_ENDIAN

Core BE8 Big-Endian state changed. Fields:

BE8 bool
: Core BE8 Big-Endian state

CORE_INFO

Static processor attributes. Only triggered by a call to DumpState(). Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ARCH_PROFILE enum
: The architecture profile of the core.

CLUSTER_ID unsigned int
: The cluster ID of this processor.

CORE_NUM unsigned int
: The number of this core in an MP processor.

DETAILED_CORE_FEATURES enum
: Detailed core features as a bitfield. Enum values are bitmasks to extract the info from the field

FPU_VERSION enum
: The VFP version implemented by the core.

MEM_ARCH enum
: The memory architecture of the core.

MIDR_EL1 unsigned int
: MIDR_EL1 value

NUM_CORES unsigned int
: The number of cores in this MP processor.

QUANTUM_SIZE unsigned int
: The default quantum size of the core.

SECURITY_FEATURES bool
: Does the core have security features?

CORE_LOADS

Processor load accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CORE_REGS

Changes of the core registers R0 to R14. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number, 0 to 14.

MODE enum
: Bank of the register accessed.

OLD_VALUE unsigned int
: The old value overwritten.

PHYS_ID enum
: The physical register accessed.

VALUE unsigned int
: The new value written to the register.

CORE_REGS64

Changes of the core registers X0..X30, SP_ELn. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: The register number, 0..30 for X0..X30, >=32 for SP_ELn.

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

CORE_STORES

Processor store accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CP14_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP14_WRITE

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CP15_WRITE

System Control Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

CPSR

CPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

OLD_VALUE unsigned int
: The old CPSR value

UNKNOWN unsigned int
: Bits within the register that have unknown value.

VALUE unsigned int
: The new CPSR value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CRYPTO_SPEC

Every crypto instruction speculatively executed.

CorePowerStateChange

Core power-up state has changed. Fields:

PU bool
: Whether the core powered up

DATA_CACHE_SET_ALLOC_TAGS

Set allocation tags from DC GVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of memory in bytes.

TAG unsigned int
: The value of the tag being set.

VADDR unsigned int
: Virtual address used for setting allocation tags.

DATA_CACHE_ZERO

Zero and invalidate cache line from DC ZVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of zero'd memory in bytes.

VADDR unsigned int
: Virtual address of the zero'd cache line.

DEBUG_EVENT

Hardware debug support event. Fields:

EVENT enum
: Description of event

VALUE unsigned int
: data value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

END_COMPILE

Compilation end. Fields:

END_OF_PAGE_COUNT unsigned int
: Number of basic blocks exited due to page boundary since START_COMPILE.

FETCHFAIL_COUNT unsigned int
: Number of basic blocks exited due to ifetch failure START_COMPILE.

INST_COUNT unsigned int
: Number of instructions compiled since START_COMPILE.

NONSEQ_COUNT unsigned int
: Number of basic blocks exited due to non-sequential instructions since START_COMPILE.

PAGE_STRADDLE_COUNT unsigned int
: Number of basic blocks exited due to unaligned instructions crossing page since START_COMPILE.

ESB

Error Synchronization Barrier hit. Fields:

ASYNC bool
: ESB is async

IMPLICIT bool
: ESB is implicit

EXCEPTION

Exception taken. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

IS_PHYSICAL bool
: Physical or Virtual exception

LR unsigned int
: The value assigned to the link register.

PC unsigned int
: The location where the exception occurred.

PREFERRED_RETURN unsigned int
: The preferred return address for the exception.

TARGET_ISET enum
: The instruction set of the exception handler code.

TARGET_PC unsigned int
: The address the exception branches to.

VECTOR enum
: The exception vector.

VECTOR_OFFSET unsigned int
: The offset of an exception vector from the base address.

EXCEPTION_END

Every exception completed.

EXCEPTION_RAISE

Every exception raised.

EXCEPTION_RETURN

Branches on leaving exception. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

EXCEPTION_RETURN_PREBRANCH

Exception return event that must occur before PC and processor state have updated.

EXCEPTION_START

Every exception started.

FIQ_TAKEN

FIQ taken exception.

FP_STATE

Floating point state. Fields:

FPSR unsigned int
: FPSR register value.

MASK unsigned int
: The mask indicating updated cumulative bits.

HLT

HLT instruction occurred. Fields:

IMM16 unsigned int
: The 16-bit immediate value encoded in HLT instruction.

INST

Every instruction executed. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DEBUG_STATE bool
: The instruction is executed in debug state.

DISASS string
: Disassembly of instruction.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The current instruction set.

ITSTATE unsigned int
: The current ITSTATE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

MODE enum
: The mode the core is in.

NS unsigned int
: The core's non-secure bit.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

OPCODE unsigned int
: The opcode of the instruction.

PADDR unsigned int
: The physical address of the instruction.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

INST_END

Every instruction completed.

INST_START

Every instruction started. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

INST_STRADDLE

Instruction straddles boundary. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

IRQ_TAKEN

IRQ taken exception.

ISB

ISB instruction occurred.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOAD_MULTIPLE_REGS_LIST

List of destination registers (not source) of a multiple registers load instruction. Fields:

REG string
: Register to add into the list

LOCAL_MONITOR

Local monitor activity. Fields:

PADDR unsigned int
: Local Monitor Address

State enum
: State of the monitor (Open/Exclusive)

MEMMAP_DEBUG_READ

Memory mapped reads to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

MEMMAP_DEBUG_WRITE

Memory mapped writes to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

MMU_TRANS

Address translation information. Fields:

ASID unsigned int
: Address space identifier.

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Hyp bool
: Entry matches in Hyp state only

INNERCACHE_RA bool
: Is the inner cache allocate on read

INNERCACHE_TRANSIENT bool
: Is the inner write-through transient

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

MEMTYPE enum
: Memory type.

NSDESC enum
: Is secure side supposed to access secure or nonsecure memory

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TRANSIENT bool
: Is the outer write-through transient

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

PADDR unsigned int
: Physical address of the access.

PAGESIZE unsigned int
: Page size as log2(size).

SH enum
: Shareability

SIDE enum
: Inst / Data.

STAGE1_PERM unsigned int
: Stage 1 permissions, [5:3] Privileged access (XWR) [2:0] User access (XWR)

STAGE2_PERM unsigned int
: Stage 2 permission mask [5:3] Privileged access (XWR) [2:0] User access (XWR)

VADDR unsigned int
: Virtual address of the access.

VMID unsigned int
: Virtual machine identifier.

XS bool
: XS Attribute

nG enum
: Flag indicating whether ASID will be matched

MMU_TTB_READ

This event is triggered by reads caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data read.

IPA unsigned int
: For stage 1, the IPA of the read.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

OUTADDR unsigned int
: The output physical address.

PADDR unsigned int
: The physical address of the read.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MMU_TTB_WRITE

This event is triggered by writes caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data written.

IPA unsigned int
: For stage 1, the IPA of the write.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

PADDR unsigned int
: The physical address of the write.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MODE_CHANGE

Mode change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: The new mode.

NON_SECURE enum
: The new security state

OLD_MODE enum
: The old mode.

NOT_TAG_CHECKED_INST

MTE Tag not checked for current instruction.

PERIODIC

Called for every quantum. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

PC unsigned int
: The address of the next instruction to be executed on this CPU.

PMU_COUNTER_OVERFLOW

PMU counter overflow. Fields:

EVENT_ID unsigned int
: Event Identifier

INDEX unsigned int
: Counter index (as selected by PMSELR).

INTERRUPT bool
: Is interrupt enabled on overflow for this counter

PREFETCH_MEMORY64

Prefetch from PRFM or PRFUM instructions. Fields:

PRFOP unsigned int
: Prefetch hint

VADDR unsigned int
: Virtual address of the location that should be prefetched.

PRELOAD_DATA

Data preload from PLD instruction. Fields:

VADDR unsigned int
: Virtual address of the data that should be preloaded.

PRELOAD_INST

Instruction preload from PLI instruction. Fields:

VADDR unsigned int
: Virtual address of the instruction that should be preloaded.

PRE_CORE_LOAD

Trace just before a core load. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_CORE_STORE

Trace just before a core store. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_TTB_READ

This event is triggered before reads caused by a translation table walk. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR unsigned int
The physical address of the read.

PSTATE_PACTIVE_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Operating mode after the transition
OLD_STATE enum
Operating mode before the transition

PSTATE_PACTIVE_OPMODE_TRANSITION_DENIED

Trace the denied pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Operating mode after the transition
OLD_STATE enum
Operating mode before the transition
REASON string
Reason for the transition being denied

PSTATE_PACTIVE_POWER_TRANSITION

Trace pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Power mode after the transition
OLD_STATE enum
Power mode before the transition

PSTATE_PACTIVE_POWER_TRANSITION_DENIED

Trace the denied pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Power mode after the transition
OLD_STATE enum
Power mode before the transition
REASON string
Reason for the transition being denied

PSTATE_PACTIVE_THREAD_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions for threads on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Operating mode after the transition
OLD_STATE enum
Operating mode before the transition

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

RUN_STATE

Run state transition. Fields:

INST_COUNT unsigned int
: Ticks count at point of transition.

NEW enum
: New run state.

OLD enum
: Old run state.

SBP_SAMPLE_COLLISION

Statistical profiling samples collided .

SBP_SAMPLE_DISCARDED

Fields:

BUFFER_POINTER unsigned int
: Value of buffer pointer

DATA unsigned int
: Data discarded

SIZE unsigned int
: Size of sample, in bytes

SBP_SAMPLE_END

Statistical profiling of an operation ended.

SBP_SAMPLE_POP

Operation which might be sampled by statistical profiling.

SBP_SAMPLE_RECORDED

Fields:

BUFFER_ADDRESS unsigned int
: Virtual Address of start of buffer write

DATA unsigned int
: Data written to the buffer

SIZE unsigned int
: Size of sample written to buffer, in bytes

SBP_SAMPLE_START

Statistical profiling of an operation started.

SBP_SAMPLE_TAKEN

Operation sampled by statistical profiling after filtering.

SEMIHOSTING_CALL

Call of a semihost function occurred. Fields:

PC unsigned int
: The program counter after the semihosting call.

SEMIHOSTING_PRECALL

About to call semihost. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

EL2 bool
: Use EL2 translation regime.

NS bool
: Is Non-Secure.

REG_WIDTH unsigned int
: The current register width in bytes.

SIGNAL

External signal state change. Fields:

SIGNAL enum
: Signal that changed

STATE bool
: Signal asserted state

SOFTWARE_STEP

Return the Debug Software Step state. Fields:

STATE enum
: Software Step state

SPSR

SPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: Which of the banked SPSR registers is written.

OLD_VALUE unsigned int
: The old SPSR value

VALUE unsigned int
: The new SPSR value

START_COMPILE

Compilation started. Fields:

VADDR unsigned int
: Instruction where compilation begins.

STG_MULTIPLE_MICRO_OPS_ADDRESS_RANGE

Address range for MTE stores with multiple uops. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

STORE_MULTIPLE_REGS_ADDRESS_RANGE

Address range for stores with multiple registers. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

SUBOPTIMAL_LDST_RETIRED

suboptimal load/store. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1: half-word, 2: word ...)

OPTIMAL_ALIGN_SELECTOR unsigned int
: Selector to decide the optimal alignment size (1: access width, 2: 4-bytes, 3: 8-bytes, ..., 12: 4-kbytes)

VADDR unsigned int
: The virtual address of the access

SYNC

Called for every synchronization, that is every quantum break. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

LOCAL_QUANTUM unsigned int
: The local quantum of this CPU.

LOCAL_TIME unsigned int
: The local time of this CPU.

SYNC_REASON enum
: Reason for sync.

SYSCALL

System call instruction executed. Fields:

IMM unsigned int
: Immediate value of the system call instruction.

TYPE enum
: System call type.

VADDR unsigned int
: Instruction that caused the system call.

SYSREG_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

opc2 unsigned int
: opcode 2

SYSREG_UPDATE32

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE64

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

UNALIGNED_LDST_RETIRED

Processor unaligned load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1=>half-word, 2->word ...)

NUMBER_OF_BEATS unsigned int
: Number of accesses (beats) in this burst (Total size in bytes is ACCESS_SIZE * NUMBER_OF_BEATS)

VADDR unsigned int
: The virtual address of the access

VFP_D_REGS

VFP/NEON D 64 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_Q_REGS

VFP/NEON Q 128 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MASK unsigned int
: Each bit indicates a write to the corresponding byte in the register. For example, 0x1 would indicate a write to VALUE[7:0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_SYS_REGS

Writes to the VFP/NEON units system registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: Which VFP system register is written.

MASK unsigned int
: The mask indicating updated cumulative bits.

OLD_VALUE unsigned int
: The register's old value overwritten.

VALUE unsigned int
: The new value written to the VFP system register.

VFP_S_REGS

VFP/NEON S 32 bit register write. Fields:

ALIASING enum
: Type of register aliasing used.

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

WAYPOINT

Signals end of basic block execution. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: Origin instructions set

IS_COND bool
: Indicates if this is a conditional waypoint.

PC unsigned int
: Origin address (or 0 if unavailable)

TAKEN bool
: Indicates if this waypoint was taken.

TARGET unsigned int
: Destination address

TARGET_ISET enum
: Destination instructions set

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

WFE_END

WFE ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFE.

WFE_EVENT_REGISTER

WFE event register status: set/clear, reason. Fields:

INST_COUNT unsigned int
: Ticks count.

REASON enum
: Reason for set/clear. Only REASON==0 (Cleared by WFE) clears the bit, all other reasons set it.

WFE_IGNORED

WFE ignored. Fields:

EVENT bool
: This WFE was ignored because the event register was set.

INST_COUNT unsigned int
: Ticks count when ignoring WFE.

TRAPPED bool
: This WFE was trapped.

WFE_START

WFE entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFE.

WFI_END

WFI ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFI.

WFI_IGNORED

WFI ignored. Fields:

DISABLED bool
: This WFI was ignored because WFI is disabled.

INST_COUNT unsigned int
: Ticks count when ignoring WFI.

REASON enum
: specifies reason why WFI trace was ignored

TRAPPED bool
: This WFI was trapped.

WFI_START

WFI entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFI.

WFI_WAKEUP

WFI wakeup. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: Ticks count when WFI wakeup occurred.

REASON enum
: Reason for wakeup.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.43 ARM_CortexR52 - trace

This section describes the trace sources.

ASYNC_MEMORY_FAULT

Context ID Register write. Fields:

FAULT unsigned int
: Fault status in ESR format

PADDR unsigned int
: Physical Address (or 0 if unavailable)

VADDR signed int
: Virtual Address (or 0 if unavailable)

ATOMIC_END_ACCESS

Bus trace access for atomics. Fields:

ACCESS_FAIL bool
: Memory access failed

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

LOAD_VALUE unsigned int
: Loaded values

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ATOMIC_START_ACCESS

Bus trace access for atomics. Fields:

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

NSDESC unsigned int
: The security state of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ArchMsg.Error.exit_code

Fields:

EXITCODE signed int
: Exit code to be returned

KIND string
: Component kind that invoked the exit code trace

REASON string
: Reason why the exit code trace was invoked (optional)

ArchMsg.Info.invalid_region_programmed

Fields:

BASE_ADDR unsigned int
: Base Address

END_ADDR unsigned int
: End Address

REGION_ID unsigned int
: Region ID

STAGE unsigned int
: MPU Stage

ArchMsg.Warning.IgnoredWarmResetRequest

Event fires when software attempts to request warm reset through write to deprecated DBGPRCR.CWRR.

ArchMsg.Warning.TarmacReconstructor

Unexpected event happened during Tarmac reconstruction. Fields:

REASON string
: Reason

ArchMsg.Warning.Unpredictably Indexed PM Event Register

Fields:

IsDirect unsigned int
: Direct Access rather than indirect

N unsigned int
: Number of Accessible Registers

SEL unsigned int
: Selector

ArchMsg.Warning.branch_to_unaligned_address

Fields:

ADDR unsigned int
: Unaligned branch address

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.cache_contents_unknown

Fields:

REGIME unsigned int
: Current translation regime

SIDE unsigned int
: 0 d-side cache or unified cache, 1 i-side

ArchMsg.Warning.dcimvac_matches_watchpoint

Watchpoints matching an AArch32 DCIMVA are implementation defined.

ArchMsg.Warning.decode_fieldmismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_fracbitsnegative

VCVT Instruction is unpredictable with a negative number of fraction bits.

ArchMsg.Warning.decode_initblock

Instruction is unpredictable in an IT block.

ArchMsg.Warning.decode_initblocknotlast

Instruction is unpredictable in an IT block when not the last.

ArchMsg.Warning.decode_invalid_condition

Instruction is unpredictable when the condition is not AL.

ArchMsg.Warning.decode_invalidfieldcombination

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_invalidvalue

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_mem_hint_unallocated

There is no memory hint allocated to this bit pattern.

ArchMsg.Warning.decode_registermatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_registermismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registeroutofrange

Float point instruction is unpredictable indexing beyond the end of the register bank.

ArchMsg.Warning.decode_sbzsbo

Reserved bits in the instruction are not canonical.

ArchMsg.Warning.decode_transactiontoobig

Attempt to perform a memory transaction of more than 128 bytes.

ArchMsg.Warning.decode_unpred_other

Instruction is unpredictable.

ArchMsg.Warning.decode_unpreduseofpc

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13inreglist

Instruction is unpredictable with R13 in register-list.

ArchMsg.Warning.decode_useofd16orhigher

Attempt to use register D16 or higher where it is not allowed.

ArchMsg.Warning.decode_writebackandbaseinlist

Instruction is unpredictable with write back when the base register is in the transfer list.

ArchMsg.Warning.decode_zeroregistersinlist

Instruction is unpredictable when no registers are to be transfered.

ArchMsg.Warning.recursive_branch

An instruction is performing a branch that targets the same instruction. If this was intended then a WFI might be more effective. Fields:

PC unsigned int
: Address of instruction causing the branch

ArchMsg.Warning.recursive_exception

An instruction has generated an exception that targets the same instruction. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CPSR unsigned int
: Processor state

PC unsigned int
: Address of instruction causing the exception

TYPE enum
: Type of exception

ArchMsg.Warning.reentrant_vector_catch

Fields:

ADDR unsigned int
: vector address

ArchMsg.Warning.reserved_it_state

Fields:

ITSTATE unsigned int
: reserved IT state

ArchMsg.Warning.tlb_contents_unknown

Translation has been enabled without invalidating the TLB. Fields:

EL unsigned int
: Exception level of translation regime

INVALIDITY unsigned int
: Which TLB may be invalid (if TLBs are separate)

NON_SECURE unsigned int
: Security state of translation regime

VMID unsigned int
: Current VMID, if applicable

ArchMsg.Warning.unknown_DLR_DSPSR

Exiting debug state with unknown DLR or DSPSR.

ArchMsg.Warning.unknown_ELR_SPSR

Returning from debug exception with unknown ELR or SPSR.

ArchMsg.Warning.unpredictable_a32_breakpoint

Fields:

ADDR1 unsigned int
: Address

ADDR2 unsigned int
: Breakpoint address

BAS unsigned int
: BAS Field

IS_ADDRESS_MISMATCH_BREAKPOINT bool
: Is Mismatch breakpoint, not a match breakpoint

N unsigned int
: Breakpoint number

OFFSET unsigned int
: Indicates which halfword actually considering into the A32 instruction

ArchMsg.Warning.unpredictable_banked_register_access

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

R unsigned int
R bit
SYSM unsigned int
SYSm

ArchMsg.Warning.unpredictable_ccfail_undef

Fields:

OPCODE unsigned int
opcode of the instruction
VADDR unsigned int
Virtual address of the instruction being executed

ArchMsg.Warning.unpredictable_ici_bits

ICI bits do not apply or are architecturally unpredictable for instruction.

ArchMsg.Warning.unpredictable_t32_breakpoint

Fields:

ADDR1 unsigned int
Address
ADDR2 unsigned int
Breakpoint address
BAS unsigned int
BAS Field
IS_ADDRESS_MISMATCH_BREAKPOINT bool
Is Mismatch breakpoint, not a match breakpoint
N unsigned int
Breakpoint number
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_t32_vector_catch

Fields:

ADDR unsigned int
Address
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_watchpoint_far

Fields:

LOWER_BOUND unsigned int
the lowest address accessed by the instruction that triggered the watchpoint
UPPER_BOUND unsigned int
the highest watchpointed address accessed by that instruction

ArchMsg.Warning.warning_AdvSIMDExpandImmUnexpectedZero

AdvSIMDExpandImm may treat this immediate value as UNPREDICTABLE.

ArchMsg.Warning.warning_ConditionalSMC

SMC instruction has UNPREDICTABLE effects when conditional when combined with traps.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_access_crosses_page_boundary

Access crosses page boundary.

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault_on_both_pages

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_spanning_different_memory

Fields:

ADDR unsigned int
: address of access crossing page

MEMTYPE_PAGE1 signed int
: First Page's memory type (0-Normal, 1-Non-Normal)

MEMTYPE_PAGE2 signed int
: Second Page's memory type (0-Normal, 1-Non-Normal)

ArchMsg.Warning.warning_access_wraps_around_memory

Fields:

WIDTH unsigned int
: Address width

ArchMsg.Warning.warning_bx_from_thumbee

Fields:

ADDR unsigned int
: Destination address

ArchMsg.Warning.warning_ccsidr_unimplemented_level

Fields:

CSSELR unsigned int
: current effective value of CSSELR

ArchMsg.Warning.warning_change_to_ns_when_tge_set

Attempting to change NS to 1 when HCR.TGE = 1.

ArchMsg.Warning.warning_contiguous_bit_check_abort

Fields:

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ArchMsg.Warning.warning_contiguous_bit_error

Fields:

CONTIG_ADDR unsigned int
: address of the TLB first read, with which this entry is expected to be contiguous

CONTIG_DATA unsigned int
: expected contents based on the entry at CONTIG_ADDR

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ENTRY_ADDR unsigned int
address of conflicting TLB entry
ENTRY_DATA unsigned int
contents of conflicting TLB entry

ArchMsg.Warning.warning_cp10_cp11_mismatch

Fields:

EL unsigned int
EL owning the control

ArchMsg.Warning.warning_cp10_cp11_reserved_value

Fields:

CP10 unsigned int
CP10 access
CP11 unsigned int
CP11 access

ArchMsg.Warning.warning_csselr_level_out_of_range

Fields:

IMPLEMENTED unsigned int
number of cache levels implemented
LEVEL unsigned int
written value of CSSELR.Level
TYPE enum
The type of cache selected by CSSELR.InD and TnD

ArchMsg.Warning.warning_debug_flow_control_bits_not_obeyed

Fields:

ERROR enum
type of error resulting
REGISTER enum
register under consideration

ArchMsg.Warning.warning_debug_register_access_during_reset

Fields:

IS_WRITE unsigned int
Write Not Read
OFFSET unsigned int
Register Offset

ArchMsg.Warning.warning_decode_cps_inconsistent_fields

The mode setting fields in the CPS instruction are inconsistent.

ArchMsg.Warning.warning_decode_invalid_state

Instruction is unpredictable in current exception-level/security state.

ArchMsg.Warning.warning_default_cacheable_mmu_on

Default cacheable is enabled (HCR.DC = 1) while MMU is enabled.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_default_cacheable_vmmu_off

Default cacheable is enabled (HCR.DC = 1) while HCR.VM = 0.

ArchMsg.Warning.warning_deprecated_wvr_bit_2_set

Fields:

N unsigned int
: index

WVR unsigned int
: DBGWVR

ArchMsg.Warning.warning_eret_while_software_step_active_pending

Missing ISB between setting MDSCR_EL1.SS and ERET.

ArchMsg.Warning.warning_exclusive_to_non_normal

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_exclusive_to_non_writeback

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_execute_from_device_memory

An attempt was made to execute from Device memory.

ArchMsg.Warning.warning_illegal_cpsr_mode

Fields:

NEW_MODE unsigned int
: New value of CPSR.M

ArchMsg.Warning.warning_illegal_srs_mode

Fields:

MODE unsigned int
: mode for Banked SP

ArchMsg.Warning.warning_implementation_defined_read_debug_register_in_SCS

Read Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_implementation_defined_sequential_security_transitions_supported

The behaviour of a sequential instruction fetches that cross from non-secure to secure memory and contain SG instruction is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_implementation_defined_stack_limit_check_supported

It is IMPLEMENTATION DEFINED whether stack pointer limit checking is performed for the this instructions. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDRESS unsigned int
: Address of instruction

OPCODE unsigned int
: instruction opcode

ArchMsg.Warning.warning_implementation_defined_write_debug_register_in_SCS

Write Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

DATA unsigned int
: data attempted to be written

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_invalid_tcr_granule

Fields:

REQUEST signed int
: page size requested (or 0 for reserved)

SUBSTITUTE signed int
: best guess available page size

TG_ID bool
: bits TG0 or TG1

ArchMsg.Warning.warning_load_multiple_user_registers_from_user_mode

An LDM(user registers) instruction executed from user mode.

ArchMsg.Warning.warning_load_pc_from_unaligned

PC loaded from an unaligned location.

ArchMsg.Warning.warning_mte_in_mem_attr_not_supported

Fields:

ATTR unsigned int
: Memory attribute

ArchMsg.Warning.warning_reserved_breakpoint_state_match

Fields:

IS_BREAKPOINT unsigned int
: Is breakpoint, not watchpoint

ArchMsg.Warning.warning_shareability

Unpredictable: combination of the 1st and 2nd stages of translation is Normal Inner Non-Cacheable, Outer Non-Cacheable.

ArchMsg.Warning.warning_software_step_set_while_enabled

MDSCR_EL1.SS set to 1 while software step debug exceptions are enabled.

ArchMsg.Warning.warning_thumb_instruction_wraps_around_memory

Thumb instruction warps around memory.

ArchMsg.Warning.warning_ttbr_sbz_bits_are_not_zero

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MSB unsigned int
x-1
TTBR enum
which TTBR register

ArchMsg.Warning.warning_unaligned_address_dbgdtrrx_write

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_address_dbgdtrtx_read

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_device

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_strongly_ordered

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unknown_sau_rnr

SAU_RNR was set to an unsupported value.

ArchMsg.Warning.warning_unpredictable_AIRCR_PRIP_and_BFHFNMINP

The effect of setting both AIRCR.BFHFNMINP and AIRCR.PRIP to 1 is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTCLRACTIVE_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTCLRACTIVE if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_and_SYSRESETREQ

When the processor is halted in Debug state, if a write to the register writes a 1 to both VECTRESET and SYSRESETREQ, the behavior is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTRESET if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_incorrect_VKEY

The value 0x05FA must be written to AIRCR.VKEY, otherwise the register write is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_EXE_RETURN_Reserved_Bit

EXC_RETURN[23:7] are reserved with the special condition that all bits should be written as one. Values other than all 1s are UNPREDICTABLE. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

EXC_RETURN unsigned int
Special PC value
SBO_MASK unsigned int
Which bits Should Be One to not UNPRED

ArchMsg.Warning.warning_unpredictable_change_to_DEMCR_MON_STEP_at_insufficient_priority

The effect of changing DEMCR.MON_STEP at an execution priority that is lower than the priority of the DebugMonitor exception is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_change_to_priority_of_active_exception

Changing the priority of an active exception is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_clear_lspact

Fields:

DRVEXC_TAKEN unsigned int
derived exception taken not pended
REACHED_LAST_STORE unsigned int
only the last faulted

ArchMsg.Warning.warning_unpredictable_dbg_exit_unaligned_dlr

Debug exit to AArch32 with DLR[0] set to 1. PC[0] can be set to 0 or to DLR[0].

ArchMsg.Warning.warning_unpredictable_exception_catch

Generation of an exception catch event is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_inconsistent_state

state on exception return is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_instruction

exception return instruction is unpredictable.

ArchMsg.Warning.warning_unpredictable_in_debug_state

Instruction is UNPREDICTABLE when executed in debug state.

ArchMsg.Warning.warning_unpredictable_mair_encoding

Fields:

ATTR unsigned int
Memory attribute

ArchMsg.Warning.warning_unpredictable_pmu_counter_access

Access to PMU counters from non-secure EL0 or EL1 is UNPREDICTABLE with MDCR_EL2.HPMN set to 0.

ArchMsg.Warning.warning_unpredictable_prioritization_breakpoint_match_vector_catch

Two events with the same priority occurred at the same instruction: Address Matching Vector Catch debug event, and Breakpoint debug event It is constrained unpredictable which is taken.

ArchMsg.Warning.warning_unpredictable_stack_selection

UNPREDICTABLE state on context switch.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_tsize_out_of_range

Fields:

TCR enum
: Which TCR

TSIZE signed int
: Value of TSize

TSIZE_MAX signed int
: The maximum allowed value

TTBR unsigned int
: Which TTBR region

ArchMsg.Warning.warning_unpredictable_unaligned_pc_as_base_register

The PC value must be word-aligned, otherwise the behavior of the instruction is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_unaligned_pop_stack

Fields:

FRAME_ADDR unsigned int
: unaligned stack frame address

ArchMsg.Warning.warning_unpredictable_vmsa_memattrib

Fields:

TEXCB unsigned int
: texcb value

TEX_REMAP bool
: use tex remap

ArchMsg.Warning.warning_unpredictable_vtcr_t0sz_sl0

Fields:

SL0 signed int
: VTCR.SL0, starting level for stage 2 translation table walks

T0SZ signed int
: VTCR.T0SZ, required input address range

ArchMsg.Warning.warning_unpredictable_write_DHCSR

Fields:

NEW unsigned int
: data attempted to be written

OLD unsigned int
: previous value

ArchMsg.Warning.warning_unpredictable_write_SHCSR

UNPREDICTABLE setting a pending bit to 1 for an exception with priority >= execution priority.

ArchMsg.Warning.warning_unsupported_access_to_memory_mapped_register

Access to memory mapped register is unsupported.

ArchMsg.Warning.warning_user_jmcr_access

Fields:

WRITE unsigned int
: access is write

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_wcr_mask_and_bas

DBGWCR_EL1.MASK is non-zero and BAS is not 0xff.

ArchMsg.Warning.warning_wcr_mask_reserved

DBGWCR_ELn.MASK is set to a reserved value.

ArchMsg.Warning.warning_wcr_non_configuous_bas

DBGWCR_EL1.BAS has non-contiguous set of ones.

ArchMsg.Warning.warning_wvr_masked_bit_not_zero

DBGWVRn_EL1.VA contains a bit masked by DBGWCRn_EL1.MASK that is not zero.

ArchMsg.Why.why_branch_target_exception

A branch target exception has occurred. Fields:

BTYPE unsigned int
: pstate.BTYPE at time branch taken

OPCODE unsigned int
: opcode at branch target

PC unsigned int
: Virtual Address of branch target

ArchMsg.Why.why_illegal_state

An illegal mode change has occured. Fields:

MESSAGE string
: The message

BRANCH_MISPREDICT

Simulating branch mispredict. Fields:

PC unsigned int
: Origin address (or 0 if unavailable)

BRA_DIR

Direct branches, to immediate address. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

BRA_INDIR

Indirect branches, perhaps to a register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR_CCFAIL

Indirect branches fail, perhaps to a register.

BRA_RET_CCFAIL

Branche return fail, perhaps to a register.

CACHE_MAINTENANCE_OP

Cache Maintenance Operation. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Specified MVA or Set/way

FUNCTION enum
: Clean / Invalidate

SCOPE enum
: Affected region

SIDE enum
: Inst / Data.

CCFAIL

Conditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCFAIL_UNC

Unconditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS

Conditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS_UNC

Unconditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CODE_CACHE_MAINT

Code cache maintenance. Fields:

END_INCL unsigned int
: Inclusive end addres for by VA or PA requests

NS bool
: Non-secure world for PA

START unsigned int
: Start address for by VA or PA requests

TYPE enum
: Request type.

COMPILE_BLOCK_END

Last instruction of basic block translated. Fields:

VADDR unsigned int
: Address of next instruction after this basic block

COMPILE_BLOCK_START

First instruction of basic block translated. Fields:

VADDR unsigned int
: Address of first instruction in basic block

COMPILE_INST

ARM instruction compiled. Fields:

DISASS string
: Disassembly of the instruction.

ISET enum
: The instruction set of this instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ITSTATE unsigned int
: The ITSTATE current for the instruction.

OPCODE unsigned int
: The opcode of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

CONTEXTIDR

Context ID Register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

NS bool
: Secure or nonsecure banked register is accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

CONTEXT_SYNC

Called for every context synchronization event. Fields:

ARCH_CSE bool
: Architectural Context Synchronization Event

SYSREG_SYNC bool
: System register synchronization event

CORE_ENDIAN

Core BE8 Big-Endian state changed. Fields:

BE8 bool
: Core BE8 Big-Endian state

CORE_INFO

Static processor attributes. Only triggered by a call to DumpState(). Fields:

ARCH_PROFILE enum
: The architecture profile of the core.

CLUSTER_ID unsigned int
: The cluster ID of this processor.

CORE_NUM unsigned int
: The number of this core in an MP processor.

DETAILED_CORE_FEATURES enum
: Detailed core features as a bitfield. Enum values are bitmasks to extract the info from the field

FPU_VERSION enum
: The VFP version implemented by the core.

MEM_ARCH enum
: The memory architecture of the core.

MIDR_EL1 unsigned int
: MIDR_EL1 value

NUM_CORES unsigned int
: The number of cores in this MP processor.

QUANTUM_SIZE unsigned int
: The default quantum size of the core.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SECURITY_FEATURES bool
: Does the core have security features?

CORE_LOADS

Processor load accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CORE_REGS

Changes of the core registers R0 to R14. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number, 0 to 14.

MODE enum
: Bank of the register accessed.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OLD_VALUE unsigned int
: The old value overwritten.

PHYS_ID enum
: The physical register accessed.

VALUE unsigned int
: The new value written to the register.

CORE_REGS64

Changes of the core registers X0..X30, SP_ELn. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: The register number, 0..30 for X0..X30, >=32 for SP_ELn.

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

CORE_STORES

Processor store accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CP14_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP14_WRITE

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ

System Coprocessor register read. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

CP15_WRITE

System Control Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

CPSR

CPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

OLD_VALUE unsigned int
: The old CPSR value

UNKNOWN unsigned int
: Bits within the register that have unknown value.

VALUE unsigned int
: The new CPSR value

CRYPTO_SPEC

Every crypto instruction speculatively executed.

CorePowerStateChange

Core power-up state has changed. Fields:

PU bool
: Whether the core powered up

DATA_CACHE_SET_ALLOC_TAGS

Set allocation tags from DC GVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of memory in bytes.

TAG unsigned int
: The value of the tag being set.

VADDR unsigned int
: Virtual address used for setting allocation tags.

DATA_CACHE_ZERO

Zero and invalidate cache line from DC ZVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of zero'd memory in bytes.

VADDR unsigned int
: Virtual address of the zero'd cache line.

DEBUG_EVENT

Hardware debug support event. Fields:

EVENT enum
: Description of event

VALUE unsigned int
: data value

END_COMPILE

Compilation end. Fields:

END_OF_PAGE_COUNT unsigned int
: Number of basic blocks exited due to page boundary since START_COMPILE.

FETCHFAIL_COUNT unsigned int
: Number of basic blocks exited due to ifetch failure START_COMPILE.

INST_COUNT unsigned int
: Number of instructions compiled since START_COMPILE.

NONSEQ_COUNT unsigned int
: Number of basic blocks exited due to non-sequential instructions since START_COMPILE.

PAGE_STRADDLE_COUNT unsigned int
: Number of basic blocks exited due to unaligned instructions crossing page since START_COMPILE.

ESB

Error Synchronization Barrier hit. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ASYNC bool
: ESB is async

IMPLICIT bool
: ESB is implicit

EXCEPTION

Exception taken. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

IS_PHYSICAL bool
: Physical or Virtual exception

LR unsigned int
: The value assigned to the link register.

PC unsigned int
: The location where the exception occurred.

PREFERRED_RETURN unsigned int
: The preferred return address for the exception.

TARGET_ISET enum
: The instruction set of the exception handler code.

TARGET_PC unsigned int
: The address the exception branches to.

VECTOR enum
: The exception vector.

VECTOR_OFFSET unsigned int
: The offset of an exception vector from the base address.

EXCEPTION_END

Every exception completed.

EXCEPTION_RAISE

Every exception raised.

EXCEPTION_RETURN

Branches on leaving exception. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

EXCEPTION_RETURN_PREBRANCH

Exception return event that must occur before PC and processor state have updated.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

EXCEPTION_START

Every exception started.

FIQ_TAKEN

FIQ taken exception.

FP_STATE

Floating point state. Fields:

FPSR unsigned int
: FPSR register value.

MASK unsigned int
: The mask indicating updated cumulative bits.

HLT

HLT instruction occurred. Fields:

IMM16 unsigned int
: The 16-bit immediate value encoded in HLT instruction.

INST

Every instruction executed. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DEBUG_STATE bool
: The instruction is executed in debug state.

DISASS string
: Disassembly of instruction.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The current instruction set.

ITSTATE unsigned int
: The current ITSTATE.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

MODE enum
: The mode the core is in.

NS unsigned int
: The core's non-secure bit.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

OPCODE unsigned int
: The opcode of the instruction.

PADDR unsigned int
: The physical address of the instruction.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the instruction.

PC unsigned int
: The address of the instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SIZE unsigned int
: The size of the instruction in bytes.

INST_END

Every instruction completed.

INST_START

Every instruction started. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

INST_STRADDLE

Instruction straddles boundary. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

IRQ_TAKEN

IRQ taken exception.

ISB

ISB instruction occurred.

LOAD_MULTIPLE_REGS_LIST

List of destination registers (not source) of a multiple registers load instruction. Fields:

REG string
: Register to add into the list

LOCAL_MONITOR

Local monitor activity. Fields:

PADDR unsigned int
: Local Monitor Address

State enum
: State of the monitor (Open/Exclusive)

MEMMAP_DEBUG_READ

Memory mapped reads to the debug registers. Fields:

ADDR unsigned int
: address

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

MEMMAP_DEBUG_WRITE

Memory mapped writes to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

MMU_TRANS

Address translation information. Fields:

ASID unsigned int
: Address space identifier.

CORE_NUM unsigned int
: Core number in a multi processor.

Hyp bool
: Entry matches in Hyp state only

INNERCACHE_RA bool
: Is the inner cache allocate on read

INNERCACHE_TRANSIENT bool
: Is the inner write-through transient

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

MEMTYPE enum
: Memory type.

NSDESC enum
: Is secure side supposed to access secure or nonsecure memory

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TRANSIENT bool
: Is the outer write-through transient

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

PADDR unsigned int
: Physical address of the access.

PAGESIZE unsigned int
: Page size as log2(size).

SH enum
: Shareability

SIDE enum
: Inst / Data.

STAGE1_PERM unsigned int
: Stage 1 permissions, [5:3] Privileged access (XWR) [2:0] User access (XWR)

STAGE2_PERM unsigned int
: Stage 2 permission mask [5:3] Privileged access (XWR) [2:0] User access (XWR)

VADDR unsigned int
: Virtual address of the access.

VMID unsigned int
: Virtual machine identifier.

XS bool
: XS Attribute

nG enum
: Flag indicating whether ASID will be matched

MMU_TTB_READ

This event is triggered by reads caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data read.

IPA unsigned int
: For stage 1, the IPA of the read.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

OUTADDR unsigned int
: The output physical address.

PADDR unsigned int
: The physical address of the read.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MMU_TTB_WRITE

This event is triggered by writes caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DATA unsigned int
: The data written.

IPA unsigned int
: For stage 1, the IPA of the write.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

PADDR unsigned int
: The physical address of the write.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MODE_CHANGE

Mode change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: The new mode.

OLD_MODE enum
: The old mode.

MPU_TRANS

Address translation information. Fields:

ACCESS_STATUS enum
: Access status(0xAA/0xBB/0xDD/0xEE=overlap/bg hit/default/bg fault)

CORE_NUM unsigned int
: Core number in a multi processor.

EXEC_PERM enum
: Execution Permission

INNERCACHE_RA bool
: Is the inner cache allocate on read

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

MEMTYPE enum
: Memory type.

NSDESC enum
: Is non secure access

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

PADDR unsigned int
: Address of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PAGESIZE unsigned int
: Page size as log2(size).

READ_PERM enum
: Read Permission

REG_NUM unsigned int
: MPU region-number 0-31

SH enum
: Shareability

SIDE enum
: Inst / Data.

STAGE unsigned int
: The translation stage

WRITE_PERM enum
: Write Permission

NOT_TAG_CHECKED_INST

MTE Tag not checked for current instruction.

PERIODIC

Called for every quantum. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

PC unsigned int
: The address of the next instruction to be executed on this CPU.

PMU_COUNTER_OVERFLOW

PMU counter overflow. Fields:

EVENT_ID unsigned int
: Event Identifier

INDEX unsigned int
: Counter index (as selected by PMSELR).

INTERRUPT bool
: Is interrupt enabled on overflow for this counter

PREFETCH_MEMORY64

Prefetch from PRFM or PRFUM instructions. Fields:

PRFOP unsigned int
: Prefetch hint

VADDR unsigned int
: Virtual address of the location that should be prefetched.

PRELOAD_DATA

Data preload from PLD instruction. Fields:

VADDR unsigned int
: Virtual address of the data that should be preloaded.

PRELOAD_INST

Instruction preload from PLI instruction. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VADDR unsigned int
Virtual address of the instruction that should be preloaded.

PRE_CORE_LOAD

Trace just before a core load. Fields:

CURRENT_TIME unsigned int
The core's current time, as simulated time plus local time.
LOCAL_TIME unsigned int
The core's local time, relative to the current quantum.

PRE_CORE_STORE

Trace just before a core store. Fields:

CURRENT_TIME unsigned int
The core's current time, as simulated time plus local time.
LOCAL_TIME unsigned int
The core's local time, relative to the current quantum.

PRE_TTB_READ

This event is triggered before reads caused by a translation table walk. Fields:

ADDR unsigned int
The physical address of the read.

PSTATE_PACTIVE_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Operating mode after the transition
OLD_STATE enum
Operating mode before the transition

PSTATE_PACTIVE_OPMODE_TRANSITION_DENIED

Trace the denied pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Operating mode after the transition
OLD_STATE enum
Operating mode before the transition
REASON string
Reason for the transition being denied

PSTATE_PACTIVE_POWER_TRANSITION

Trace pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Power mode after the transition
OLD_STATE enum
Power mode before the transition

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PSTATE_PACTIVE_POWER_TRANSITION_DENIED

Trace the denied pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_THREAD_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions for threads on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

RUN_STATE

Run state transition. Fields:

INST_COUNT unsigned int
: Ticks count at point of transition.

NEW enum
: New run state.

OLD enum
: Old run state.

SEMIHOSTING_CALL

Call of a semihost function occurred. Fields:

PC unsigned int
: The program counter after the semihosting call.

SEMIHOSTING_PRECALL

About to call semihost. Fields:

EL2 bool
: Use EL2 translation regime.

NS bool
: Is Non-Secure.

REG_WIDTH unsigned int
: The current register width in bytes.

SIGNAL

External signal state change. Fields:

SIGNAL enum
: Signal that changed

STATE bool
: Signal asserted state

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SOFTWARE_STEP

Return the Debug Software Step state. Fields:

STATE enum
: Software Step state

SPSR

SPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: Which of the banked SPSR registers is written.

OLD_VALUE unsigned int
: The old SPSR value

VALUE unsigned int
: The new SPSR value

START_COMPILE

Compilation started. Fields:

VADDR unsigned int
: Instruction where compilation begins.

STG_MULTIPLE_MICRO_OPS_ADDRESS_RANGE

Address range for MTE stores with multiple uops. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

STORE_MULTIPLE_REGS_ADDRESS_RANGE

Address range for stores with multiple registers. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

SUBOPTIMAL_LDST_RETIRED

suboptimal load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1: half-word, 2: word ...)

OPTIMAL_ALIGN_SELECTOR unsigned int
: Selector to decide the optimal alignment size (1: access width, 2: 4-bytes, 3: 8-bytes, ..., 12: 4-kbytes)

VADDR unsigned int
: The virtual address of the access

SYNC

Called for every synchronization, that is every quantum break. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: The instruction count of this CPU.

LOCAL_QUANTUM unsigned int
: The local quantum of this CPU.

LOCAL_TIME unsigned int
: The local time of this CPU.

SYNC_REASON enum
: Reason for sync.

SYSCALL

System call instruction executed. Fields:

IMM unsigned int
: Immediate value of the system call instruction.

TYPE enum
: System call type.

VADDR unsigned int
: Instruction that caused the system call.

SYSREG_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE32

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE64

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

UNALIGNED_LDST_RETIRED

Processor unaligned load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1=>half-word, 2->word ...)

NUMBER_OF_BEATS unsigned int
: Number of accesses (beats) in this burst (Total size in bytes is ACCESS_SIZE * NUMBER_OF_BEATS)

VADDR unsigned int
: The virtual address of the access

VFP_D_REGS

VFP/NEON D 64 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_Q_REGS

VFP/NEON Q 128 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding byte in the register. For example, 0x1 would indicate a write to VALUE[7:0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_SYS_REGS

Writes to the VFP/NEON units system registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: Which VFP system register is written.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MASK unsigned int
: The mask indicating updated cumulative bits.

OLD_VALUE unsigned int
: The register's old value overwritten.

VALUE unsigned int
: The new value written to the VFP system register.

VFP_S_REGS

VFP/NEON S 32 bit register write. Fields:

ALIASING enum
: Type of register aliasing used.

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

WAYPOINT

Signals end of basic block execution. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: Origin instructions set

IS_COND bool
: Indicates if this is a conditional waypoint.

PC unsigned int
: Origin address (or 0 if unavailable)

TAKEN bool
: Indicates if this waypoint was taken.

TARGET unsigned int
: Destination address

TARGET_ISET enum
: Destination instructions set

WFE_END

WFE ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFE.

WFE_EVENT_REGISTER

WFE event register status: set/clear, reason. Fields:

INST_COUNT unsigned int
: Ticks count.

REASON enum
: Reason for set/clear. Only REASON==0 (Cleared by WFE) clears the bit, all other reasons set it.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

WFE_IGNORED

WFE ignored. Fields:

EVENT bool
: This WFE was ignored because the event register was set.

INST_COUNT unsigned int
: Ticks count when ignoring WFE.

TRAPPED bool
: This WFE was trapped.

WFE_START

WFE entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFE.

WFI_END

WFI ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFI.

WFI_IGNORED

WFI ignored. Fields:

DISABLED bool
: This WFI was ignored because WFI is disabled.

INST_COUNT unsigned int
: Ticks count when ignoring WFI.

REASON enum
: specifies reason why WFI trace was ignored

TRAPPED bool
: This WFI was trapped.

WFI_START

WFI entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFI.

WFI_WAKEUP

WFI wakeup. Fields:

INST_COUNT unsigned int
: Ticks count when WFI wakeup occurred.

REASON enum
: Reason for wakeup.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.44 ARM_Neoverse-E1 - trace

This section describes the trace sources.

ASYNC_MEMORY_FAULT

Context ID Register write. Fields:

FAULT unsigned int
: Fault status in ESR format

PADDR unsigned int
: Physical Address (or 0 if unavailable)

VADDR signed int
: Virtual Address (or 0 if unavailable)

ATOMIC_END_ACCESS

Bus trace access for atomics. Fields:

ACCESS_FAIL bool
: Memory access failed

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

LOAD_VALUE unsigned int
: Loaded values

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ATOMIC_START_ACCESS

Bus trace access for atomics. Fields:

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

NSDESC unsigned int
: The security state of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ArchMsg.Error.exit_code

Fields:

EXITCODE signed int
: Exit code to be returned

KIND string
: Component kind that invoked the exit code trace

REASON string
: Reason why the exit code trace was invoked (optional)

ArchMsg.Warning.IgnoredWarmResetRequest

Event fires when software attempts to request warm reset through write to deprecated DBGPRCR.CWRR.

ArchMsg.Warning.RAS_ErrselrUnpredictableState

ERRSELR is in unpredictable state due to value greater than number_of_error_records written to it.

ArchMsg.Warning.RAS_UnpredictableStateWrite

Fields:

REG_NAME string
: Name of register being written.

ArchMsg.Warning.TarmacReconstructor

Unexpected event happened during Tarmac reconstruction. Fields:

REASON string
: Reason

ArchMsg.Warning.Unpredictably Indexed PM Event Register

Fields:

IsDirect unsigned int
: Direct Access rather than indirect

N unsigned int
: Number of Accessible Registers

SEL unsigned int
: Selector

ArchMsg.Warning.branch_to_unaligned_address

Fields:

ADDR unsigned int
: Unaligned branch address

ArchMsg.Warning.cache_contents_unknown

A cache has been activated without being invalidated and may contain UNKNOWN content. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

REGIME unsigned int
: Current translation regime

SIDE unsigned int
: 0 d-side cache or unified cache, 1 i-side

ArchMsg.Warning.dcimvac_matches_watchpoint

Watchpoints matching an AArch32 DCIMVA are implementation defined.

ArchMsg.Warning.decode_fieldmismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_fracbitsnegative

VCVT Instruction is unpredictable with a negative number of fraction bits.

ArchMsg.Warning.decode_initblock

Instruction is unpredictable in an IT block.

ArchMsg.Warning.decode_initblocknotlast

Instruction is unpredictable in an IT block when not the last.

ArchMsg.Warning.decode_invalid_condition

Instruction is unpredictable when the condition is not AL.

ArchMsg.Warning.decode_invalidfieldcombination

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_invalidvalue

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_mem_hint_unallocated

There is no memory hint allocated to this bit pattern.

ArchMsg.Warning.decode_registermatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_registermismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registeroutofrange

Float point instruction is unpredictable indexing beyond the end of the register bank.

ArchMsg.Warning.decode_sbzsbo

Reserved bits in the instruction are not canonical.

ArchMsg.Warning.decode_transactiontoobig

Attempt to perform a memory transaction of more than 128 bytes.

ArchMsg.Warning.decode_unpred_other

Instruction is unpredictable.

ArchMsg.Warning.decode_unpreduseofpc

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13inreglist

Instruction is unpredictable with R13 in register-list.

ArchMsg.Warning.decode_useofd16orhigher

Attempt to use register D16 or higher where it is not allowed.

ArchMsg.Warning.decode_writebackandbaseinlist

Instruction is unpredictable with write back when the base register is in the transfer list.

ArchMsg.Warning.decode_zeroregistersinlist

Instruction is unpredictable when no registers are to be transfered.

ArchMsg.Warning.recursive_branch

An instruction is performing a branch that targets the same instruction. If this was intended then a WFI might be more effective. Fields:

PC unsigned int
: Address of instruction causing the branch

ArchMsg.Warning.recursive_exception

An instruction has generated an exception that targets the same instruction. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CPSR unsigned int
: Processor state

PC unsigned int
: Address of instruction causing the exception

TYPE enum
: Type of exception

ArchMsg.Warning.reentrant_vector_catch

Fields:

ADDR unsigned int
: vector address

ArchMsg.Warning.reserved_it_state

Fields:

ITSTATE unsigned int
: reserved IT state

ArchMsg.Warning.tlb_contents_unknown

Translation has been enabled without invalidating the TLB. Fields:

EL unsigned int
: Exception level of translation regime

INVALIDITY unsigned int
: Which TLB may be invalid (if TLBs are separate)

NON_SECURE unsigned int
: Security state of translation regime

VMID unsigned int
: Current VMID, if applicable

ArchMsg.Warning.unknown_DLR_DSPSR

Exiting debug state with unknown DLR or DSPSR.

ArchMsg.Warning.unknown_ELR_SPSR

Returning from debug exception with unknown ELR or SPSR.

ArchMsg.Warning.unpredictable_a32_breakpoint

Fields:

ADDR1 unsigned int
: Address

ADDR2 unsigned int
: Breakpoint address

BAS unsigned int
: BAS Field

IS_ADDRESS_MISMATCH_BREAKPOINT bool
: Is Mismatch breakpoint, not a match breakpoint

N unsigned int
: Breakpoint number

OFFSET unsigned int
: Indicates which halfword actually considering into the A32 instruction

ArchMsg.Warning.unpredictable_banked_register_access

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

R unsigned int
: R bit

SYSM unsigned int
: SYSm

ArchMsg.Warning.unpredictable_ccfail_undef

Fields:

OPCODE unsigned int
: opcode of the instruction

VADDR unsigned int
: Virtual address of the instruction being executed

ArchMsg.Warning.unpredictable_ici_bits

ICI bits do not apply or are architecturally unpredictable for instruction.

ArchMsg.Warning.unpredictable_t32_breakpoint

Fields:

ADDR1 unsigned int
: Address

ADDR2 unsigned int
: Breakpoint address

BAS unsigned int
: BAS Field

IS_ADDRESS_MISMATCH_BREAKPOINT bool
: Is Mismatch breakpoint, not a match breakpoint

N unsigned int
: Breakpoint number

OFFSET unsigned int
: Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_t32_vector_catch

Fields:

ADDR unsigned int
: Address

OFFSET unsigned int
: Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_watchpoint_far

Fields:

LOWER_BOUND unsigned int
: the lowest address accessed by the instruction that triggered the watchpoint

UPPER_BOUND unsigned int
: the highest watchpointed address accessed by that instruction

ArchMsg.Warning.warning_AdvSIMDExpandImmUnexpectedZero

AdvSIMDExpandImm may treat this immediate value as UNPREDICTABLE.

ArchMsg.Warning.warning_ConditionalSMC

SMC instruction has UNPREDICTABLE effects when conditional when combined with traps.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_access_crosses_page_boundary

Access crosses page boundary.

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault_on_both_pages

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_spanning_different_memory

Fields:

ADDR `unsigned int`
: address of access crossing page

MEMTYPE_PAGE1 `signed int`
: First Page's memory type (0-Normal, 1-Non-Normal)

MEMTYPE_PAGE2 `signed int`
: Second Page's memory type (0-Normal, 1-Non-Normal)

ArchMsg.Warning.warning_access_wraps_around_memory

Fields:

WIDTH `unsigned int`
: Address width

ArchMsg.Warning.warning_bx_from_thumbee

Fields:

ADDR `unsigned int`
: Destination address

ArchMsg.Warning.warning_ccsidr_unimplemented_level

Fields:

CSSELR `unsigned int`
: current effective value of CSSELR

ArchMsg.Warning.warning_change_to_ns_when_tge_set

Attempting to change NS to 1 when HCR.TGE = 1.

ArchMsg.Warning.warning_contiguous_bit_check_abort

Fields:

ENTRY_ADDR `unsigned int`
: address of conflicting TLB entry

ArchMsg.Warning.warning_contiguous_bit_error

Fields:

CONTIG_ADDR `unsigned int`
: address of the TLB first read, with which this entry is expected to be contiguous

CONTIG_DATA `unsigned int`
: expected contents based on the entry at CONTIG_ADDR

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ENTRY_DATA unsigned int
: contents of conflicting TLB entry

ArchMsg.Warning.warning_cp10_cp11_mismatch

Fields:

EL unsigned int
: EL owning the control

ArchMsg.Warning.warning_cp10_cp11_reserved_value

Fields:

CP10 unsigned int
: CP10 access

CP11 unsigned int
: CP11 access

ArchMsg.Warning.warning_csselr_level_out_of_range

Fields:

IMPLEMENTED unsigned int
: number of cache levels implemented

LEVEL unsigned int
: written value of CSSELR.Level

TYPE enum
: The type of cache selected by CSSELR.InD and TnD

ArchMsg.Warning.warning_debug_flow_control_bits_not_obeyed

Fields:

ERROR enum
: type of error resulting

REGISTER enum
: register under consideration

ArchMsg.Warning.warning_debug_register_access_during_reset

Fields:

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

ArchMsg.Warning.warning_decode_cps_inconsistent_fields

The mode setting fields in the CPS instruction are inconsistent.

ArchMsg.Warning.warning_decode_invalid_state

Instruction is unpredictable in current exception-level/security state.

ArchMsg.Warning.warning_default_cacheable_mmu_on

Default cacheable is enabled (HCR.DC = 1) while MMU is enabled.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_default_cacheable_vmmu_off

Default cacheable is enabled (HCR.DC = 1) while HCR.VM = 0.

ArchMsg.Warning.warning_deprecated_wvr_bit_2_set

Fields:

N unsigned int
: index

WVR unsigned int
: DBGWVR

ArchMsg.Warning.warning_eret_while_software_step_active_pending

Missing ISB between setting MDSCR_EL1.SS and ERET.

ArchMsg.Warning.warning_exclusive_to_non_normal

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_exclusive_to_non_writeback

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_execute_from_device_memory

An attempt was made to execute from Device memory.

ArchMsg.Warning.warning_illegal_cpsr_mode

Fields:

NEW_MODE unsigned int
: New value of CPSR.M

ArchMsg.Warning.warning_illegal_srs_mode

Fields:

MODE unsigned int
: mode for Banked SP

ArchMsg.Warning.warning_implementation_defined_read_debug_register_in_SCS

Read Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_implementation_defined_sequential_security_transitions_supported

The behaviour of a sequential instruction fetches that cross from non-secure to secure memory and contain SG instruction is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_implementation_defined_stack_limit_check_supported

It is IMPLEMENTATION DEFINED whether stack pointer limit checking is performed for the this instructions. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDRESS unsigned int
: Address of instruction

OPCODE unsigned int
: instruction opcode

ArchMsg.Warning.warning_implementation_defined_write_debug_register_in_SCS

Write Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

DATA unsigned int
: data attempted to be written

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_invalid_tcr_granule

Fields:

REQUEST signed int
: page size requested (or 0 for reserved)

SUBSTITUTE signed int
: best guess available page size

TG_ID bool
: bits TG0 or TG1

ArchMsg.Warning.warning_load_multiple_user_registers_from_user_mode

An LDM(user registers) instruction executed from user mode.

ArchMsg.Warning.warning_load_pc_from_unaligned

PC loaded from an unaligned location.

ArchMsg.Warning.warning_mte_in_mem_attr_not_supported

Fields:

ATTR unsigned int
: Memory attribute

ArchMsg.Warning.warning_reserved_breakpoint_state_match

Fields:

IS_BREAKPOINT unsigned int
: Is breakpoint, not watchpoint

ArchMsg.Warning.warning_shareability

Unpredictable: combination of the 1st and 2nd stages of translation is Normal Inner Non-Cacheable, Outer Non-Cacheable.

ArchMsg.Warning.warning_software_step_set_while_enabled

MDSCR_EL1.SS set to 1 while software step debug exceptions are enabled.

ArchMsg.Warning.warning_thumb_instruction_wraps_around_memory

Thumb instruction warps around memory.

ArchMsg.Warning.warning_ttbr_sbz_bits_are_not_zero

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MSB unsigned int
x-1
TTBR enum
which TTBR register

ArchMsg.Warning.warning_unaligned_address_dbgdtrrx_write

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_address_dbgdtrtx_read

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_device

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_strongly_ordered

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unknown_sau_rnr

SAU_RNR was set to an unsupported value.

ArchMsg.Warning.warning_unpredictable_AIRCR_PRIP_and_BFHFNMINP

The effect of setting both AIRCR.BFHFNMINP and AIRCR.PRIP to 1 is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTCLRACTIVE_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTCLRACTIVE if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_and_SYSRESETREQ

When the processor is halted in Debug state, if a write to the register writes a 1 to both VECTRESET and SYSRESETREQ, the behavior is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTRESET if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_incorrect_VKEY

The value 0x05FA must be written to AIRCR.VKEY, otherwise the register write is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_EXE_RETURN_Reserved_Bit

EXC_RETURN[23:7] are reserved with the special condition that all bits should be written as one. Values other than all 1s are UNPREDICTABLE. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

EXC_RETURN unsigned int
: Special PC value

SBO_MASK unsigned int
: Which bits Should Be One to not UNPRED

ArchMsg.Warning.warning_unpredictable_change_to_DEMCR_MON_STEP_at_insufficient_priority

The effect of changing DEMCR.MON_STEP at an execution priority that is lower than the priority of the DebugMonitor exception is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_change_to_priority_of_active_exception

Changing the priority of an active exception is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_clear_lspact

Fields:

DRVEXC_TAKEN unsigned int
: derived exception taken not pended

REACHED_LAST_STORE unsigned int
: only the last faulted

ArchMsg.Warning.warning_unpredictable_dbg_exit_unaligned_dlr

Debug exit to AArch32 with DLR[0] set to 1. PC[0] can be set to 0 or to DLR[0].

ArchMsg.Warning.warning_unpredictable_exception_catch

Generation of an exception catch event is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_inconsistent_state

state on exception return is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_instruction

exception return instruction is unpredictable.

ArchMsg.Warning.warning_unpredictable_in_debug_state

Instruction is UNPREDICTABLE when executed in debug state.

ArchMsg.Warning.warning_unpredictable_mair_encoding

Fields:

ATTR unsigned int
: Memory attribute

ArchMsg.Warning.warning_unpredictable_pmu_counter_access

Access to PMU counters from non-secure EL0 or EL1 is UNPREDICTABLE with MDCR_EL2.HPMN set to 0.

ArchMsg.Warning.warning_unpredictable_prioritization_breakpoint_match_vector_catch

Two events with the same priority occurred at the same instruction: Address Matching Vector Catch debug event, and Breakpoint debug event It is constrained unpredictable which is taken.

ArchMsg.Warning.warning_unpredictable_stack_selection

UNPREDICTABLE state on context switch.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_tsize_out_of_range

Fields:

TCR enum
: Which TCR

TSIZE signed int
: Value of TSize

TSIZE_MAX signed int
: The maximum allowed value

TTBR unsigned int
: Which TTBR region

ArchMsg.Warning.warning_unpredictable_unaligned_pc_as_base_register

The PC value must be word-aligned, otherwise the behavior of the instruction is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_unaligned_pop_stack

Fields:

FRAME_ADDR unsigned int
: unaligned stack frame address

ArchMsg.Warning.warning_unpredictable_vmsa_memattrib

Fields:

TEXCB unsigned int
: texcb value

TEX_REMAP bool
: use tex remap

ArchMsg.Warning.warning_unpredictable_vtcr_t0sz_sl0

Fields:

SL0 signed int
: VTCR.SL0, starting level for stage 2 translation table walks

T0SZ signed int
: VTCR.T0SZ, required input address range

ArchMsg.Warning.warning_unpredictable_write_DHCSR

Fields:

NEW unsigned int
: data attempted to be written

OLD unsigned int
: previous value

ArchMsg.Warning.warning_unpredictable_write_SHCSR

UNPREDICTABLE setting a pending bit to 1 for an exception with priority >= execution priority.

ArchMsg.Warning.warning_unsupported_access_to_memory_mapped_register

Access to memory mapped register is unsupported.

ArchMsg.Warning.warning_user_jmcr_access

Fields:

WRITE unsigned int
: access is write

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_wcr_mask_and_bas

DBGWCR_EL1.MASK is non-zero and BAS is not 0xff.

ArchMsg.Warning.warning_wcr_mask_reserved

DBGWCR_ELn.MASK is set to a reserved value.

ArchMsg.Warning.warning_wcr_non_configuous_bas

DBGWCR_EL1.BAS has non-contiguous set of ones.

ArchMsg.Warning.warning_wvr_masked_bit_not_zero

DBGWVRn_EL1.VA contains a bit masked by DBGWCRn_EL1.MASK that is not zero.

ArchMsg.Why.why_branch_target_exception

A branch target exception has occurred. Fields:

BTYPE unsigned int
: pstate.BTYPE at time branch taken

OPCODE unsigned int
: opcode at branch target

PC unsigned int
: Virtual Address of branch target

ArchMsg.Why.why_illegal_state

An illegal mode change has occured. Fields:

MESSAGE string
: The message

BRANCH_MISPREDICT

Simulating branch mispredict. Fields:

PC unsigned int
: Origin address (or 0 if unavailable)

BRA_DIR

Direct branches, to immediate address. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

BRA_INDIR

Indirect branches, perhaps to a register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR_CCFAIL

Indirect branches fail, perhaps to a register.

BRA_RET_CCFAIL

Branche return fail, perhaps to a register.

CACHE_MAINTENANCE_OP

Cache Maintenance Operation. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Specified MVA or Set/way

FUNCTION enum
: Clean / Invalidate

SCOPE enum
: Affected region

SIDE enum
: Inst / Data.

CCFAIL

Conditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCFAIL_UNC

Unconditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS

Conditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS_UNC

Unconditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CHECKPOINT_MESSAGE

Report error messages from the checkpointing process. Fields:

message string
: Message contents.

CHECKPOINT_RESTORE_END

Checkpoint restore completed.

CHECKPOINT_RESTORE_START

Checkpoint restore about to start.

CHECKPOINT_SAVE_END

Checkpoint save completed.

CHECKPOINT_SAVE_START

Checkpoint save about to start.

CODE_CACHE_MAINT

Code cache maintenance. Fields:

END_INCL unsigned int
: Inclusive end addres for by VA or PA requests

NS bool
: Non-secure world for PA

START unsigned int
: Start address for by VA or PA requests

TYPE enum
: Request type.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

COMPILE_BLOCK_END

Last instruction of basic block translated. Fields:

VADDR unsigned int
: Address of next instruction after this basic block

COMPILE_BLOCK_START

First instruction of basic block translated. Fields:

VADDR unsigned int
: Address of first instruction in basic block

COMPILE_INST

ARM instruction compiled. Fields:

DISASS string
: Disassembly of the instruction.

ISET enum
: The instruction set of this instruction.

ITSTATE unsigned int
: The ITSTATE current for the instruction.

OPCODE unsigned int
: The opcode of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

CONTEXTIDR

Context ID Register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

NS bool
: Secure or nonsecure banked register is accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

CONTEXT_SYNC

Called for every context synchronization event. Fields:

ARCH_CSE bool
: Architectural Context Synchronization Event

SYSREG_SYNC bool
: System register synchronization event

CORE_ENDIAN

Core BE8 Big-Endian state changed. Fields:

BE8 bool
: Core BE8 Big-Endian state

CORE_INFO

Static processor attributes. Only triggered by a call to DumpState(). Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ARCH_PROFILE enum
: The architecture profile of the core.

CLUSTER_ID unsigned int
: The cluster ID of this processor.

CORE_NUM unsigned int
: The number of this core in an MP processor.

DETAILED_CORE_FEATURES enum
: Detailed core features as a bitfield. Enum values are bitmasks to extract the info from the field

FPU_VERSION enum
: The VFP version implemented by the core.

MEM_ARCH enum
: The memory architecture of the core.

MIDR_EL1 unsigned int
: MIDR_EL1 value

NUM_CORES unsigned int
: The number of cores in this MP processor.

QUANTUM_SIZE unsigned int
: The default quantum size of the core.

SECURITY_FEATURES bool
: Does the core have security features?

CORE_LOADS

Processor load accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CORE_REGS

Changes of the core registers R0 to R14. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number, 0 to 14.

MODE enum
: Bank of the register accessed.

OLD_VALUE unsigned int
: The old value overwritten.

PHYS_ID enum
: The physical register accessed.

VALUE unsigned int
: The new value written to the register.

CORE_REGS64

Changes of the core registers X0..X30, SP_ELn. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: The register number, 0..30 for X0..X30, >=32 for SP_ELn.

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

CORE_STORES

Processor store accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CP14_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP14_WRITE

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CP15_WRITE

System Control Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

CPSR

CPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

OLD_VALUE unsigned int
: The old CPSR value

UNKNOWN unsigned int
: Bits within the register that have unknown value.

VALUE unsigned int
: The new CPSR value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CRYPTO_SPEC

Every crypto instruction speculatively executed.

CorePowerStateChange

Core power-up state has changed. Fields:

PU bool
: Whether the core powered up

DATA_CACHE_SET_ALLOC_TAGS

Set allocation tags from DC GVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of memory in bytes.

TAG unsigned int
: The value of the tag being set.

VADDR unsigned int
: Virtual address used for setting allocation tags.

DATA_CACHE_ZERO

Zero and invalidate cache line from DC ZVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of zero'd memory in bytes.

VADDR unsigned int
: Virtual address of the zero'd cache line.

DEBUG_EVENT

Hardware debug support event. Fields:

EVENT enum
: Description of event

VALUE unsigned int
: data value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

END_COMPILE

Compilation end. Fields:

END_OF_PAGE_COUNT unsigned int
: Number of basic blocks exited due to page boundary since START_COMPILE.

FETCHFAIL_COUNT unsigned int
: Number of basic blocks exited due to ifetch failure START_COMPILE.

INST_COUNT unsigned int
: Number of instructions compiled since START_COMPILE.

NONSEQ_COUNT unsigned int
: Number of basic blocks exited due to non-sequential instructions since START_COMPILE.

PAGE_STRADDLE_COUNT unsigned int
: Number of basic blocks exited due to unaligned instructions crossing page since START_COMPILE.

ESB

Error Synchronization Barrier hit. Fields:

ASYNC bool
: ESB is async

IMPLICIT bool
: ESB is implicit

EXCEPTION

Exception taken. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

IS_PHYSICAL bool
: Physical or Virtual exception

LR unsigned int
: The value assigned to the link register.

PC unsigned int
: The location where the exception occurred.

PREFERRED_RETURN unsigned int
: The preferred return address for the exception.

TARGET_ISET enum
: The instruction set of the exception handler code.

TARGET_PC unsigned int
: The address the exception branches to.

VECTOR enum
: The exception vector.

VECTOR_OFFSET unsigned int
: The offset of an exception vector from the base address.

EXCEPTION_END

Every exception completed.

EXCEPTION_RAISE

Every exception raised.

EXCEPTION_RETURN

Branches on leaving exception. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

EXCEPTION_RETURN_PREBRANCH

Exception return event that must occur before PC and processor state have updated.

EXCEPTION_START

Every exception started.

FIQ_TAKEN

FIQ taken exception.

FP_STATE

Floating point state. Fields:

FPSR unsigned int
: FPSR register value.

MASK unsigned int
: The mask indicating updated cumulative bits.

HLT

HLT instruction occurred. Fields:

IMM16 unsigned int
: The 16-bit immediate value encoded in HLT instruction.

INST

Every instruction executed. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DEBUG_STATE bool
: The instruction is executed in debug state.

DISASS string
: Disassembly of instruction.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The current instruction set.

ITSTATE unsigned int
: The current ITSTATE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

MODE enum
: The mode the core is in.

NS unsigned int
: The core's non-secure bit.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

OPCODE unsigned int
: The opcode of the instruction.

PADDR unsigned int
: The physical address of the instruction.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

INST_END

Every instruction completed.

INST_START

Every instruction started. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

INST_STRADDLE

Instruction straddles boundary. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

IRQ_TAKEN

IRQ taken exception.

ISB

ISB instruction occurred.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOAD_MULTIPLE_REGS_LIST

List of destination registers (not source) of a multiple registers load instruction. Fields:

REG string
: Register to add into the list

LOCAL_MONITOR

Local monitor activity. Fields:

PADDR unsigned int
: Local Monitor Address

State enum
: State of the monitor (Open/Exclusive)

MEMMAP_DEBUG_READ

Memory mapped reads to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

MEMMAP_DEBUG_WRITE

Memory mapped writes to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

MMU_TRANS

Address translation information. Fields:

ASID unsigned int
: Address space identifier.

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Hyp bool
: Entry matches in Hyp state only

INNERCACHE_RA bool
: Is the inner cache allocate on read

INNERCACHE_TRANSIENT bool
: Is the inner write-through transient

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

MEMTYPE enum
: Memory type.

NSDESC enum
: Is secure side supposed to access secure or nonsecure memory

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TRANSIENT bool
: Is the outer write-through transient

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

PADDR unsigned int
: Physical address of the access.

PAGESIZE unsigned int
: Page size as log2(size).

SH enum
: Shareability

SIDE enum
: Inst / Data.

STAGE1_PERM unsigned int
: Stage 1 permissions, [5:3] Privileged access (XWR) [2:0] User access (XWR)

STAGE2_PERM unsigned int
: Stage 2 permission mask [5:3] Privileged access (XWR) [2:0] User access (XWR)

VADDR unsigned int
: Virtual address of the access.

VMID unsigned int
: Virtual machine identifier.

XS bool
: XS Attribute

nG enum
: Flag indicating whether ASID will be matched

MMU_TTB_READ

This event is triggered by reads caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data read.

IPA unsigned int
: For stage 1, the IPA of the read.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

OUTADDR unsigned int
: The output physical address.

PADDR unsigned int
: The physical address of the read.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MMU_TTB_WRITE

This event is triggered by writes caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data written.

IPA unsigned int
: For stage 1, the IPA of the write.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

PADDR unsigned int
: The physical address of the write.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MODE_CHANGE

Mode change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: The new mode.

NON_SECURE enum
: The new security state

OLD_MODE enum
: The old mode.

NOT_TAG_CHECKED_INST

MTE Tag not checked for current instruction.

PERIODIC

Called for every quantum. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

PC unsigned int
: The address of the next instruction to be executed on this CPU.

PMU_COUNTER_OVERFLOW

PMU counter overflow. Fields:

EVENT_ID unsigned int
: Event Identifier

INDEX unsigned int
: Counter index (as selected by PMSELR).

INTERRUPT bool
: Is interrupt enabled on overflow for this counter

PREFETCH_MEMORY64

Prefetch from PRFM or PRFUM instructions. Fields:

PRFOP unsigned int
: Prefetch hint

VADDR unsigned int
: Virtual address of the location that should be prefetched.

PRELOAD_DATA

Data preload from PLD instruction. Fields:

VADDR unsigned int
: Virtual address of the data that should be preloaded.

PRELOAD_INST

Instruction preload from PLI instruction. Fields:

VADDR unsigned int
: Virtual address of the instruction that should be preloaded.

PRE_CORE_LOAD

Trace just before a core load. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_CORE_STORE

Trace just before a core store. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_TTB_READ

This event is triggered before reads caused by a translation table walk. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR unsigned int
The physical address of the read.

PSTATE_PACTIVE_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Operating mode after the transition
OLD_STATE enum
Operating mode before the transition

PSTATE_PACTIVE_OPMODE_TRANSITION_DENIED

Trace the denied pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Operating mode after the transition
OLD_STATE enum
Operating mode before the transition
REASON string
Reason for the transition being denied

PSTATE_PACTIVE_POWER_TRANSITION

Trace pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Power mode after the transition
OLD_STATE enum
Power mode before the transition

PSTATE_PACTIVE_POWER_TRANSITION_DENIED

Trace the denied pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Power mode after the transition
OLD_STATE enum
Power mode before the transition
REASON string
Reason for the transition being denied

PSTATE_PACTIVE_THREAD_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions for threads on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Operating mode after the transition
OLD_STATE enum
Operating mode before the transition

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

RUN_STATE

Run state transition. Fields:

INST_COUNT unsigned int
: Ticks count at point of transition.

NEW enum
: New run state.

OLD enum
: Old run state.

SEMIHOSTING_CALL

Call of a semihost function occurred. Fields:

PC unsigned int
: The program counter after the semihosting call.

SEMIHOSTING_PRECALL

About to call semihost. Fields:

EL2 bool
: Use EL2 translation regime.

NS bool
: Is Non-Secure.

REG_WIDTH unsigned int
: The current register width in bytes.

SIGNAL

External signal state change. Fields:

SIGNAL enum
: Signal that changed

STATE bool
: Signal asserted state

SOFTWARE_STEP

Return the Debug Software Step state. Fields:

STATE enum
: Software Step state

SPSR

SPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: Which of the banked SPSR registers is written.

OLD_VALUE unsigned int
: The old SPSR value

VALUE unsigned int
: The new SPSR value

START_COMPILE

Compilation started. Fields:

VADDR unsigned int
: Instruction where compilation begins.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

STG_MULTIPLE_MICRO_OPS_ADDRESS_RANGE

Address range for MTE stores with multiple uops. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

STORE_MULTIPLE_REGS_ADDRESS_RANGE

Address range for stores with multiple registers. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

SUBOPTIMAL_LDST_RETIRED

suboptimal load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1: half-word, 2: word ...)

OPTIMAL_ALIGN_SELECTOR unsigned int
: Selector to decide the optimal alignment size (1: access width, 2: 4-bytes, 3: 8-bytes, ..., 12: 4-kbytes)

VADDR unsigned int
: The virtual address of the access

SYNC

Called for every synchronization, that is every quantum break. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

LOCAL_QUANTUM unsigned int
: The local quantum of this CPU.

LOCAL_TIME unsigned int
: The local time of this CPU.

SYNC_REASON enum
: Reason for sync.

SYSCALL

System call instruction executed. Fields:

IMM unsigned int
: Immediate value of the system call instruction.

TYPE enum
: System call type.

VADDR unsigned int
: Instruction that caused the system call.

SYSREG_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE32

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE64

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

UNALIGNED_LDST_RETIRED

Processor unaligned load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1=>half-word, 2->word ...)

NUMBER_OF_BEATS unsigned int
: Number of accesses (beats) in this burst (Total size in bytes is ACCESS_SIZE * NUMBER_OF_BEATS)

VADDR unsigned int
: The virtual address of the access

VFP_D_REGS

VFP/NEON D 64 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_Q_REGS

VFP/NEON Q 128 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding byte in the register. For example, 0x1 would indicate a write to VALUE[7:0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_SYS_REGS

Writes to the VFP/NEON units system registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: Which VFP system register is written.

MASK unsigned int
: The mask indicating updated cumulative bits.

OLD_VALUE unsigned int
: The register's old value overwritten.

VALUE unsigned int
: The new value written to the VFP system register.

VFP_S_REGS

VFP/NEON S 32 bit register write. Fields:

ALIASING enum
: Type of register aliasing used.

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VALUE unsigned int
: The new value written to the register.

WAYPOINT

Signals end of basic block execution. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: Origin instructions set

IS_COND bool
: Indicates if this is a conditional waypoint.

PC unsigned int
: Origin address (or 0 if unavailable)

TAKEN bool
: Indicates if this waypoint was taken.

TARGET unsigned int
: Destination address

TARGET_ISET enum
: Destination instructions set

WFE_END

WFE ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFE.

WFE_EVENT_REGISTER

WFE event register status: set/clear, reason. Fields:

INST_COUNT unsigned int
: Ticks count.

REASON enum
: Reason for set/clear. Only REASON==0 (Cleared by WFE) clears the bit, all other reasons set it.

WFE_IGNORED

WFE ignored. Fields:

EVENT bool
: This WFE was ignored because the event register was set.

INST_COUNT unsigned int
: Ticks count when ignoring WFE.

TRAPPED bool
: This WFE was trapped.

WFE_START

WFE entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFE.

WFI_END

WFI ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFI.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

WFI_IGNORED

WFI ignored. Fields:

DISABLED bool
: This WFI was ignored because WFI is disabled.

INST_COUNT unsigned int
: Ticks count when ignoring WFI.

REASON enum
: specifies reason why WFI trace was ignored

TRAPPED bool
: This WFI was trapped.

WFI_START

WFI entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFI.

WFI_WAKEUP

WFI wakeup. Fields:

INST_COUNT unsigned int
: Ticks count when WFI wakeup occurred.

REASON enum
: Reason for wakeup.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.45 ARM_Neoverse-N1 - trace

This section describes the trace sources.

ASYNC_MEMORY_FAULT

Context ID Register write. Fields:

FAULT unsigned int
: Fault status in ESR format

PADDR unsigned int
: Physical Address (or 0 if unavailable)

VADDR signed int
: Virtual Address (or 0 if unavailable)

ATOMIC_END_ACCESS

Bus trace access for atomics. Fields:

ACCESS_FAIL bool
: Memory access failed

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

LOAD_VALUE unsigned int
: Loaded values

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ATOMIC_START_ACCESS

Bus trace access for atomics. Fields:

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

NSDESC unsigned int
: The security state of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ArchMsg.Error.exit_code

Fields:

EXITCODE signed int
: Exit code to be returned

KIND string
: Component kind that invoked the exit code trace

REASON string
: Reason why the exit code trace was invoked (optional)

ArchMsg.Warning.IgnoredWarmResetRequest

Event fires when software attempts to request warm reset through write to deprecated DBGPRCR.CWRR.

ArchMsg.Warning.RAS_ErrselrUnpredictableState

ERRSELR is in unpredictable state due to value greater than number_of_error_records written to it.

ArchMsg.Warning.RAS_UnpredictableStateWrite

Fields:

REG_NAME string
: Name of register being written.

ArchMsg.Warning.TarmacReconstructor

Unexpected event happened during Tarmac reconstruction. Fields:

REASON string
: Reason

ArchMsg.Warning.Unpredictably Indexed PM Event Register

Fields:

IsDirect unsigned int
: Direct Access rather than indirect

N unsigned int
: Number of Accessible Registers

SEL unsigned int
: Selector

ArchMsg.Warning.branch_to_unaligned_address

Fields:

ADDR unsigned int
: Unaligned branch address

ArchMsg.Warning.cache_contents_unknown

A cache has been activated without being invalidated and may contain UNKNOWN content. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

REGIME unsigned int
: Current translation regime

SIDE unsigned int
: 0 d-side cache or unified cache, 1 i-side

ArchMsg.Warning.dcimvac_matches_watchpoint

Watchpoints matching an AArch32 DCIMVA are implementation defined.

ArchMsg.Warning.decode_fieldmismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_fracbitsnegative

VCVT Instruction is unpredictable with a negative number of fraction bits.

ArchMsg.Warning.decode_initblock

Instruction is unpredictable in an IT block.

ArchMsg.Warning.decode_initblocknotlast

Instruction is unpredictable in an IT block when not the last.

ArchMsg.Warning.decode_invalid_condition

Instruction is unpredictable when the condition is not AL.

ArchMsg.Warning.decode_invalidfieldcombination

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_invalidvalue

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_mem_hint_unallocated

There is no memory hint allocated to this bit pattern.

ArchMsg.Warning.decode_registermatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_registermismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registeroutofrange

Float point instruction is unpredictable indexing beyond the end of the register bank.

ArchMsg.Warning.decode_sbzsbo

Reserved bits in the instruction are not canonical.

ArchMsg.Warning.decode_transactiontoobig

Attempt to perform a memory transaction of more than 128 bytes.

ArchMsg.Warning.decode_unpred_other

Instruction is unpredictable.

ArchMsg.Warning.decode_unpreduseofpc

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13inreglist

Instruction is unpredictable with R13 in register-list.

ArchMsg.Warning.decode_useofd16orhigher

Attempt to use register D16 or higher where it is not allowed.

ArchMsg.Warning.decode_writebackandbaseinlist

Instruction is unpredictable with write back when the base register is in the transfer list.

ArchMsg.Warning.decode_zeroregistersinlist

Instruction is unpredictable when no registers are to be transfered.

ArchMsg.Warning.recursive_branch

An instruction is performing a branch that targets the same instruction. If this was intended then a WFI might be more effective. Fields:

PC unsigned int
: Address of instruction causing the branch

ArchMsg.Warning.recursive_exception

An instruction has generated an exception that targets the same instruction. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CPSR unsigned int
Processor state
PC unsigned int
Address of instruction causing the exception
TYPE enum
Type of exception

ArchMsg.Warning.reentrant_vector_catch

Fields:

ADDR unsigned int
vector address

ArchMsg.Warning.reserved_it_state

Fields:

ITSTATE unsigned int
reserved IT state

ArchMsg.Warning.tlb_contents_unknown

Translation has been enabled without invalidating the TLB. Fields:

EL unsigned int
Exception level of translation regime
INVALIDITY unsigned int
Which TLB may be invalid (if TLBs are separate)
NON_SECURE unsigned int
Security state of translation regime
VMID unsigned int
Current VMID, if applicable

ArchMsg.Warning.unknown_DLR_DSPSR

Exiting debug state with unknown DLR or DSPSR.

ArchMsg.Warning.unknown_ELR_SPSR

Returning from debug exception with unknown ELR or SPSR.

ArchMsg.Warning.unpredictable_a32_breakpoint

Fields:

ADDR1 unsigned int
Address
ADDR2 unsigned int
Breakpoint address
BAS unsigned int
BAS Field
IS_ADDRESS_MISMATCH_BREAKPOINT bool
Is Mismatch breakpoint, not a match breakpoint
N unsigned int
Breakpoint number
OFFSET unsigned int
Indicates which halfword actually considering into the A32 instruction

ArchMsg.Warning.unpredictable_banked_register_access

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

R unsigned int
R bit
SYSM unsigned int
SYSm

ArchMsg.Warning.unpredictable_ccfail_undef

Fields:

OPCODE unsigned int
opcode of the instruction
VADDR unsigned int
Virtual address of the instruction being executed

ArchMsg.Warning.unpredictable_ici_bits

ICI bits do not apply or are architecturally unpredictable for instruction.

ArchMsg.Warning.unpredictable_t32_breakpoint

Fields:

ADDR1 unsigned int
Address
ADDR2 unsigned int
Breakpoint address
BAS unsigned int
BAS Field
IS_ADDRESS_MISMATCH_BREAKPOINT bool
Is Mismatch breakpoint, not a match breakpoint
N unsigned int
Breakpoint number
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_t32_vector_catch

Fields:

ADDR unsigned int
Address
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_watchpoint_far

Fields:

LOWER_BOUND unsigned int
the lowest address accessed by the instruction that triggered the watchpoint
UPPER_BOUND unsigned int
the highest watchpointed address accessed by that instruction

ArchMsg.Warning.warning_AdvSIMDExpandImmUnexpectedZero

AdvSIMDExpandImm may treat this immediate value as UNPREDICTABLE.

ArchMsg.Warning.warning_ConditionalSMC

SMC instruction has UNPREDICTABLE effects when conditional when combined with traps.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_access_crosses_page_boundary

Access crosses page boundary.

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault_on_both_pages

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_spanning_different_memory

Fields:

ADDR `unsigned int`
: address of access crossing page

MEMTYPE_PAGE1 `signed int`
: First Page's memory type (0-Normal, 1-Non-Normal)

MEMTYPE_PAGE2 `signed int`
: Second Page's memory type (0-Normal, 1-Non-Normal)

ArchMsg.Warning.warning_access_wraps_around_memory

Fields:

WIDTH `unsigned int`
: Address width

ArchMsg.Warning.warning_bx_from_thumbee

Fields:

ADDR `unsigned int`
: Destination address

ArchMsg.Warning.warning_ccsidr_unimplemented_level

Fields:

CSSELR `unsigned int`
: current effective value of CSSELR

ArchMsg.Warning.warning_change_to_ns_when_tge_set

Attempting to change NS to 1 when HCR.TGE = 1.

ArchMsg.Warning.warning_contiguous_bit_check_abort

Fields:

ENTRY_ADDR `unsigned int`
: address of conflicting TLB entry

ArchMsg.Warning.warning_contiguous_bit_error

Fields:

CONTIG_ADDR `unsigned int`
: address of the TLB first read, with which this entry is expected to be contiguous

CONTIG_DATA `unsigned int`
: expected contents based on the entry at CONTIG_ADDR

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ENTRY_ADDR unsigned int
: address of conflicting TLB entry

ENTRY_DATA unsigned int
: contents of conflicting TLB entry

ArchMsg.Warning.warning_cp10_cp11_mismatch

Fields:

EL unsigned int
: EL owning the control

ArchMsg.Warning.warning_cp10_cp11_reserved_value

Fields:

CP10 unsigned int
: CP10 access

CP11 unsigned int
: CP11 access

ArchMsg.Warning.warning_csselr_level_out_of_range

Fields:

IMPLEMENTED unsigned int
: number of cache levels implemented

LEVEL unsigned int
: written value of CSSELR.Level

TYPE enum
: The type of cache selected by CSSELR.InD and TnD

ArchMsg.Warning.warning_debug_flow_control_bits_not_obeyed

Fields:

ERROR enum
: type of error resulting

REGISTER enum
: register under consideration

ArchMsg.Warning.warning_debug_register_access_during_reset

Fields:

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

ArchMsg.Warning.warning_decode_cps_inconsistent_fields

The mode setting fields in the CPS instruction are inconsistent.

ArchMsg.Warning.warning_decode_invalid_state

Instruction is unpredictable in current exception-level/security state.

ArchMsg.Warning.warning_default_cacheable_mmu_on

Default cacheable is enabled (HCR.DC = 1) while MMU is enabled.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_default_cacheable_vmmu_off

Default cacheable is enabled (HCR.DC = 1) while HCR.VM = 0.

ArchMsg.Warning.warning_deprecated_wvr_bit_2_set

Fields:

N unsigned int
: index

WVR unsigned int
: DBGWVR

ArchMsg.Warning.warning_eret_while_software_step_active_pending

Missing ISB between setting MDSCR_EL1.SS and ERET.

ArchMsg.Warning.warning_exclusive_to_non_normal

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_exclusive_to_non_writeback

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_execute_from_device_memory

An attempt was made to execute from Device memory.

ArchMsg.Warning.warning_illegal_cpsr_mode

Fields:

NEW_MODE unsigned int
: New value of CPSR.M

ArchMsg.Warning.warning_illegal_srs_mode

Fields:

MODE unsigned int
: mode for Banked SP

ArchMsg.Warning.warning_implementation_defined_read_debug_register_in_SCS

Read Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_implementation_defined_sequential_security_transitions_supported

The behaviour of a sequential instruction fetches that cross from non-secure to secure memory and contain SG instruction is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_implementation_defined_stack_limit_check_supported

It is IMPLEMENTATION DEFINED whether stack pointer limit checking is performed for the this instructions. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDRESS unsigned int
: Address of instruction

OPCODE unsigned int
: instruction opcode

ArchMsg.Warning.warning_implementation_defined_write_debug_register_in_SCS

Write Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

DATA unsigned int
: data attempted to be written

PPB_OFFSET unsigned int
: debug register offset

ArchMsg.Warning.warning_invalid_tcr_granule

Fields:

REQUEST signed int
: page size requested (or 0 for reserved)

SUBSTITUTE signed int
: best guess available page size

TG_ID bool
: bits TG0 or TG1

ArchMsg.Warning.warning_load_multiple_user_registers_from_user_mode

An LDM(user registers) instruction executed from user mode.

ArchMsg.Warning.warning_load_pc_from_unaligned

PC loaded from an unaligned location.

ArchMsg.Warning.warning_mte_in_mem_attr_not_supported

Fields:

ATTR unsigned int
: Memory attribute

ArchMsg.Warning.warning_reserved_breakpoint_state_match

Fields:

IS_BREAKPOINT unsigned int
: Is breakpoint, not watchpoint

ArchMsg.Warning.warning_shareability

Unpredictable: combination of the 1st and 2nd stages of translation is Normal Inner Non-Cacheable, Outer Non-Cacheable.

ArchMsg.Warning.warning_software_step_set_while_enabled

MDSCR_EL1.SS set to 1 while software step debug exceptions are enabled.

ArchMsg.Warning.warning_thumb_instruction_wraps_around_memory

Thumb instruction warps around memory.

ArchMsg.Warning.warning_ttbr_sbz_bits_are_not_zero

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MSB unsigned int
x-1
TTBR enum
which TTBR register

ArchMsg.Warning.warning_unaligned_address_dbgdtrrx_write

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_address_dbgdtrtx_read

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_device

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_strongly_ordered

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unknown_sau_rnr

SAU_RNR was set to an unsupported value.

ArchMsg.Warning.warning_unpredictable_AIRCR_PRIP_and_BFHFNMINP

The effect of setting both AIRCR.BFHFNMINP and AIRCR.PRIP to 1 is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTCLRACTIVE_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTCLRACTIVE if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_and_SYSRESETREQ

When the processor is halted in Debug state, if a write to the register writes a 1 to both VECTRESET and SYSRESETREQ, the behavior is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTRESET if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_incorrect_VKEY

The value 0x05FA must be written to AIRCR.VKEY, otherwise the register write is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_EXE_RETURN_Reserved_Bit

EXC_RETURN[23:7] are reserved with the special condition that all bits should be written as one. Values other than all 1s are UNPREDICTABLE. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

EXC_RETURN unsigned int
Special PC value
SBO_MASK unsigned int
Which bits Should Be One to not UNPRED

ArchMsg.Warning.warning_unpredictable_change_to_DEMCR_MON_STEP_at_insufficient_priority

The effect of changing DEMCR.MON_STEP at an execution priority that is lower than the priority of the DebugMonitor exception is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_change_to_priority_of_active_exception

Changing the priority of an active exception is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_clear_lspact

Fields:

DRVEXC_TAKEN unsigned int
derived exception taken not pended
REACHED_LAST_STORE unsigned int
only the last faulted

ArchMsg.Warning.warning_unpredictable_dbg_exit_unaligned_dlr

Debug exit to AArch32 with DLR[0] set to 1. PC[0] can be set to 0 or to DLR[0].

ArchMsg.Warning.warning_unpredictable_exception_catch

Generation of an exception catch event is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_inconsistent_state

state on exception return is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_instruction

exception return instruction is unpredictable.

ArchMsg.Warning.warning_unpredictable_in_debug_state

Instruction is UNPREDICTABLE when executed in debug state.

ArchMsg.Warning.warning_unpredictable_mair_encoding

Fields:

ATTR unsigned int
Memory attribute

ArchMsg.Warning.warning_unpredictable_pmu_counter_access

Access to PMU counters from non-secure EL0 or EL1 is UNPREDICTABLE with MDCR_EL2.HPMN set to 0.

ArchMsg.Warning.warning_unpredictable_prioritization_breakpoint_match_vector_catch

Two events with the same priority occurred at the same instruction: Address Matching Vector Catch debug event, and Breakpoint debug event It is constrained unpredictable which is taken.

ArchMsg.Warning.warning_unpredictable_stack_selection

UNPREDICTABLE state on context switch.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_tsize_out_of_range

Fields:

TCR enum
: Which TCR

TSIZE signed int
: Value of TSize

TSIZE_MAX signed int
: The maximum allowed value

TTBR unsigned int
: Which TTBR region

ArchMsg.Warning.warning_unpredictable_unaligned_pc_as_base_register

The PC value must be word-aligned, otherwise the behavior of the instruction is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_unaligned_pop_stack

Fields:

FRAME_ADDR unsigned int
: unaligned stack frame address

ArchMsg.Warning.warning_unpredictable_vmsa_memattrib

Fields:

TEXCB unsigned int
: texcb value

TEX_REMAP bool
: use tex remap

ArchMsg.Warning.warning_unpredictable_vtcr_t0sz_sl0

Fields:

SL0 signed int
: VTCR.SL0, starting level for stage 2 translation table walks

T0SZ signed int
: VTCR.T0SZ, required input address range

ArchMsg.Warning.warning_unpredictable_write_DHCSR

Fields:

NEW unsigned int
: data attempted to be written

OLD unsigned int
: previous value

ArchMsg.Warning.warning_unpredictable_write_SHCSR

UNPREDICTABLE setting a pending bit to 1 for an exception with priority >= execution priority.

ArchMsg.Warning.warning_unsupported_access_to_memory_mapped_register

Access to memory mapped register is unsupported.

ArchMsg.Warning.warning_user_jmcr_access

Fields:

WRITE unsigned int
: access is write

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_wcr_mask_and_bas

DBGWCR_EL1.MASK is non-zero and BAS is not 0xff.

ArchMsg.Warning.warning_wcr_mask_reserved

DBGWCR_ELn.MASK is set to a reserved value.

ArchMsg.Warning.warning_wcr_non_configuous_bas

DBGWCR_EL1.BAS has non-contiguous set of ones.

ArchMsg.Warning.warning_wvr_masked_bit_not_zero

DBGWVRn_EL1.VA contains a bit masked by DBGWCRn_EL1.MASK that is not zero.

ArchMsg.Why.why_branch_target_exception

A branch target exception has occurred. Fields:

BTYPE unsigned int
: pstate.BTYPE at time branch taken

OPCODE unsigned int
: opcode at branch target

PC unsigned int
: Virtual Address of branch target

ArchMsg.Why.why_illegal_state

An illegal mode change has occured. Fields:

MESSAGE string
: The message

BRANCH_MISPREDICT

Simulating branch mispredict. Fields:

PC unsigned int
: Origin address (or 0 if unavailable)

BRA_DIR

Direct branches, to immediate address. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

BRA_INDIR

Indirect branches, perhaps to a register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR_CCFAIL

Indirect branches fail, perhaps to a register.

BRA_RET_CCFAIL

Branche return fail, perhaps to a register.

CACHE_MAINTENANCE_OP

Cache Maintenance Operation. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Specified MVA or Set/way

FUNCTION enum
: Clean / Invalidate

SCOPE enum
: Affected region

SIDE enum
: Inst / Data.

CCFAIL

Conditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCFAIL_UNC

Unconditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS

Conditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS_UNC

Unconditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CHECKPOINT_MESSAGE

Report error messages from the checkpointing process. Fields:

message string
: Message contents.

CHECKPOINT_RESTORE_END

Checkpoint restore completed.

CHECKPOINT_RESTORE_START

Checkpoint restore about to start.

CHECKPOINT_SAVE_END

Checkpoint save completed.

CHECKPOINT_SAVE_START

Checkpoint save about to start.

CODE_CACHE_MAINT

Code cache maintenance. Fields:

END_INCL unsigned int
: Inclusive end addres for by VA or PA requests

NS bool
: Non-secure world for PA

START unsigned int
: Start address for by VA or PA requests

TYPE enum
: Request type.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

COMPILE_BLOCK_END

Last instruction of basic block translated. Fields:

VADDR unsigned int
: Address of next instruction after this basic block

COMPILE_BLOCK_START

First instruction of basic block translated. Fields:

VADDR unsigned int
: Address of first instruction in basic block

COMPILE_INST

ARM instruction compiled. Fields:

DISASS string
: Disassembly of the instruction.

ISET enum
: The instruction set of this instruction.

ITSTATE unsigned int
: The ITSTATE current for the instruction.

OPCODE unsigned int
: The opcode of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

CONTEXTIDR

Context ID Register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

NS bool
: Secure or nonsecure banked register is accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

CONTEXT_SYNC

Called for every context synchronization event. Fields:

ARCH_CSE bool
: Architectural Context Synchronization Event

SYSREG_SYNC bool
: System register synchronization event

CORE_ENDIAN

Core BE8 Big-Endian state changed. Fields:

BE8 bool
: Core BE8 Big-Endian state

CORE_INFO

Static processor attributes. Only triggered by a call to DumpState(). Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ARCH_PROFILE enum
: The architecture profile of the core.

CLUSTER_ID unsigned int
: The cluster ID of this processor.

CORE_NUM unsigned int
: The number of this core in an MP processor.

DETAILED_CORE_FEATURES enum
: Detailed core features as a bitfield. Enum values are bitmasks to extract the info from the field

FPU_VERSION enum
: The VFP version implemented by the core.

MEM_ARCH enum
: The memory architecture of the core.

MIDR_EL1 unsigned int
: MIDR_EL1 value

NUM_CORES unsigned int
: The number of cores in this MP processor.

QUANTUM_SIZE unsigned int
: The default quantum size of the core.

SECURITY_FEATURES bool
: Does the core have security features?

CORE_LOADS

Processor load accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CORE_REGS

Changes of the core registers R0 to R14. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number, 0 to 14.

MODE enum
: Bank of the register accessed.

OLD_VALUE unsigned int
: The old value overwritten.

PHYS_ID enum
: The physical register accessed.

VALUE unsigned int
: The new value written to the register.

CORE_REGS64

Changes of the core registers X0..X30, SP_ELn. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: The register number, 0..30 for X0..X30, >=32 for SP_ELn.

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

CORE_STORES

Processor store accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CP14_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP14_WRITE

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP14 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CP15_WRITE

System Control Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc1 unsigned int
: opcode 1

opc2 unsigned int
: opcode 2

CP15_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

NS bool
: Secure or nonsecure banked register is accessed.

REG_NAME string
: Name of the CP15 register accessed.

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

CPSR

CPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

OLD_VALUE unsigned int
: The old CPSR value

UNKNOWN unsigned int
: Bits within the register that have unknown value.

VALUE unsigned int
: The new CPSR value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CRYPTO_SPEC

Every crypto instruction speculatively executed.

CorePowerStateChange

Core power-up state has changed. Fields:

PU bool
: Whether the core powered up

DATA_CACHE_SET_ALLOC_TAGS

Set allocation tags from DC GVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of memory in bytes.

TAG unsigned int
: The value of the tag being set.

VADDR unsigned int
: Virtual address used for setting allocation tags.

DATA_CACHE_ZERO

Zero and invalidate cache line from DC ZVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of zero'd memory in bytes.

VADDR unsigned int
: Virtual address of the zero'd cache line.

DEBUG_EVENT

Hardware debug support event. Fields:

EVENT enum
: Description of event

VALUE unsigned int
: data value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

END_COMPILE

Compilation end. Fields:

END_OF_PAGE_COUNT unsigned int
: Number of basic blocks exited due to page boundary since START_COMPILE.

FETCHFAIL_COUNT unsigned int
: Number of basic blocks exited due to ifetch failure START_COMPILE.

INST_COUNT unsigned int
: Number of instructions compiled since START_COMPILE.

NONSEQ_COUNT unsigned int
: Number of basic blocks exited due to non-sequential instructions since START_COMPILE.

PAGE_STRADDLE_COUNT unsigned int
: Number of basic blocks exited due to unaligned instructions crossing page since START_COMPILE.

ESB

Error Synchronization Barrier hit. Fields:

ASYNC bool
: ESB is async

IMPLICIT bool
: ESB is implicit

EXCEPTION

Exception taken. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

IS_PHYSICAL bool
: Physical or Virtual exception

LR unsigned int
: The value assigned to the link register.

PC unsigned int
: The location where the exception occurred.

PREFERRED_RETURN unsigned int
: The preferred return address for the exception.

TARGET_ISET enum
: The instruction set of the exception handler code.

TARGET_PC unsigned int
: The address the exception branches to.

VECTOR enum
: The exception vector.

VECTOR_OFFSET unsigned int
: The offset of an exception vector from the base address.

EXCEPTION_END

Every exception completed.

EXCEPTION_RAISE

Every exception raised.

EXCEPTION_RETURN

Branches on leaving exception. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

EXCEPTION_RETURN_PREBRANCH

Exception return event that must occur before PC and processor state have updated.

EXCEPTION_START

Every exception started.

FIQ_TAKEN

FIQ taken exception.

FP_STATE

Floating point state. Fields:

FPSR unsigned int
: FPSR register value.

MASK unsigned int
: The mask indicating updated cumulative bits.

HLT

HLT instruction occurred. Fields:

IMM16 unsigned int
: The 16-bit immediate value encoded in HLT instruction.

INST

Every instruction executed. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DEBUG_STATE bool
: The instruction is executed in debug state.

DISASS string
: Disassembly of instruction.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The current instruction set.

ITSTATE unsigned int
: The current ITSTATE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

MODE enum
: The mode the core is in.

NS unsigned int
: The core's non-secure bit.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

OPCODE unsigned int
: The opcode of the instruction.

PADDR unsigned int
: The physical address of the instruction.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

INST_END

Every instruction completed.

INST_START

Every instruction started. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

INST_STRADDLE

Instruction straddles boundary. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

IRQ_TAKEN

IRQ taken exception.

ISB

ISB instruction occurred.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOAD_MULTIPLE_REGS_LIST

List of destination registers (not source) of a multiple registers load instruction. Fields:

REG string
: Register to add into the list

LOCAL_MONITOR

Local monitor activity. Fields:

PADDR unsigned int
: Local Monitor Address

State enum
: State of the monitor (Open/Exclusive)

MEMMAP_DEBUG_READ

Memory mapped reads to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

MEMMAP_DEBUG_WRITE

Memory mapped writes to the debug registers. Fields:

ADDR unsigned int
: address

CORE_NUM unsigned int
: Core number in a multi processor.

EXT bool
: Whether access is from an external device (such as the DAP)

NS enum
: Secure state of the access

REG_NAME string
: Name of the debug register accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

MMU_TRANS

Address translation information. Fields:

ASID unsigned int
: Address space identifier.

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Hyp bool
: Entry matches in Hyp state only

INNERCACHE_RA bool
: Is the inner cache allocate on read

INNERCACHE_TRANSIENT bool
: Is the inner write-through transient

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

MEMTYPE enum
: Memory type.

NSDESC enum
: Is secure side supposed to access secure or nonsecure memory

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TRANSIENT bool
: Is the outer write-through transient

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

PADDR unsigned int
: Physical address of the access.

PAGESIZE unsigned int
: Page size as log2(size).

SH enum
: Shareability

SIDE enum
: Inst / Data.

STAGE1_PERM unsigned int
: Stage 1 permissions, [5:3] Privileged access (XWR) [2:0] User access (XWR)

STAGE2_PERM unsigned int
: Stage 2 permission mask [5:3] Privileged access (XWR) [2:0] User access (XWR)

VADDR unsigned int
: Virtual address of the access.

VMID unsigned int
: Virtual machine identifier.

XS bool
: XS Attribute

nG enum
: Flag indicating whether ASID will be matched

MMU_TTB_READ

This event is triggered by reads caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data read.

IPA unsigned int
: For stage 1, the IPA of the read.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

OUTADDR unsigned int
: The output physical address.

PADDR unsigned int
: The physical address of the read.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MMU_TTB_WRITE

This event is triggered by writes caused by a translation table walk. Fields:

ABORTED bool
: Was the walk successful.

DATA unsigned int
: The data written.

IPA unsigned int
: For stage 1, the IPA of the write.

LEVEL unsigned int
: Translation table level

LPAE bool
: Is this for an LPAE translation

NSmem bool
: Non secure memory access

NSreq bool
: Non secure request

PADDR unsigned int
: The physical address of the write.

SIDE enum
: Inst / Data.

STAGE unsigned int
: Translation stage

MODE_CHANGE

Mode change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: The new mode.

NON_SECURE enum
: The new security state

OLD_MODE enum
: The old mode.

NOT_TAG_CHECKED_INST

MTE Tag not checked for current instruction.

PERIODIC

Called for every quantum. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

PC unsigned int
: The address of the next instruction to be executed on this CPU.

PMU_COUNTER_OVERFLOW

PMU counter overflow. Fields:

EVENT_ID unsigned int
: Event Identifier

INDEX unsigned int
: Counter index (as selected by PMSELR).

INTERRUPT bool
: Is interrupt enabled on overflow for this counter

PREFETCH_MEMORY64

Prefetch from PRFM or PRFUM instructions. Fields:

PRFOP unsigned int
: Prefetch hint

VADDR unsigned int
: Virtual address of the location that should be prefetched.

PRELOAD_DATA

Data preload from PLD instruction. Fields:

VADDR unsigned int
: Virtual address of the data that should be preloaded.

PRELOAD_INST

Instruction preload from PLI instruction. Fields:

VADDR unsigned int
: Virtual address of the instruction that should be preloaded.

PRE_CORE_LOAD

Trace just before a core load. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_CORE_STORE

Trace just before a core store. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_TTB_READ

This event is triggered before reads caused by a translation table walk. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR unsigned int
The physical address of the read.

PSTATE_PACTIVE_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Operating mode after the transition
OLD_STATE enum
Operating mode before the transition

PSTATE_PACTIVE_OPMODE_TRANSITION_DENIED

Trace the denied pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Operating mode after the transition
OLD_STATE enum
Operating mode before the transition
REASON string
Reason for the transition being denied

PSTATE_PACTIVE_POWER_TRANSITION

Trace pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Power mode after the transition
OLD_STATE enum
Power mode before the transition

PSTATE_PACTIVE_POWER_TRANSITION_DENIED

Trace the denied pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Power mode after the transition
OLD_STATE enum
Power mode before the transition
REASON string
Reason for the transition being denied

PSTATE_PACTIVE_THREAD_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions for threads on PChannel. Fields:

DEVPACTIVE unsigned int
Integer representation of DEVPACTIVE[] signals
NEW_STATE enum
Operating mode after the transition
OLD_STATE enum
Operating mode before the transition

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

RUN_STATE

Run state transition. Fields:

INST_COUNT unsigned int
: Ticks count at point of transition.

NEW enum
: New run state.

OLD enum
: Old run state.

SBP_SAMPLE_COLLISION

Statistical profiling samples collided .

SBP_SAMPLE_DISCARDED

Fields:

BUFFER_POINTER unsigned int
: Value of buffer pointer

DATA unsigned int
: Data discarded

SIZE unsigned int
: Size of sample, in bytes

SBP_SAMPLE_END

Statistical profiling of an operation ended.

SBP_SAMPLE_POP

Operation which might be sampled by statistical profiling.

SBP_SAMPLE_RECORDED

Fields:

BUFFER_ADDRESS unsigned int
: Virtual Address of start of buffer write

DATA unsigned int
: Data written to the buffer

SIZE unsigned int
: Size of sample written to buffer, in bytes

SBP_SAMPLE_START

Statistical profiling of an operation started.

SBP_SAMPLE_TAKEN

Operation sampled by statistical profiling after filtering.

SEMIHOSTING_CALL

Call of a semihost function occurred. Fields:

PC unsigned int
: The program counter after the semihosting call.

SEMIHOSTING_PRECALL

About to call semihost. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

EL2 bool
: Use EL2 translation regime.

NS bool
: Is Non-Secure.

REG_WIDTH unsigned int
: The current register width in bytes.

SIGNAL

External signal state change. Fields:

SIGNAL enum
: Signal that changed

STATE bool
: Signal asserted state

SOFTWARE_STEP

Return the Debug Software Step state. Fields:

STATE enum
: Software Step state

SPSR

SPSR change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: Which of the banked SPSR registers is written.

OLD_VALUE unsigned int
: The old SPSR value

VALUE unsigned int
: The new SPSR value

START_COMPILE

Compilation started. Fields:

VADDR unsigned int
: Instruction where compilation begins.

STG_MULTIPLE_MICRO_OPS_ADDRESS_RANGE

Address range for MTE stores with multiple uops. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

STORE_MULTIPLE_REGS_ADDRESS_RANGE

Address range for stores with multiple registers. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

SUBOPTIMAL_LDST_RETIRED

suboptimal load/store. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1: half-word, 2: word ...)

OPTIMAL_ALIGN_SELECTOR unsigned int
: Selector to decide the optimal alignment size (1: access width, 2: 4-bytes, 3: 8-bytes, ..., 12: 4-kbytes)

VADDR unsigned int
: The virtual address of the access

SYNC

Called for every synchronization, that is every quantum break. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

LOCAL_QUANTUM unsigned int
: The local quantum of this CPU.

LOCAL_TIME unsigned int
: The local time of this CPU.

SYNC_REASON enum
: Reason for sync.

SYSCALL

System call instruction executed. Fields:

IMM unsigned int
: Immediate value of the system call instruction.

TYPE enum
: System call type.

VADDR unsigned int
: Instruction that caused the system call.

SYSREG_READ64

System Coprocessor register read. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The value read.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

opc2 unsigned int
: opcode 2

SYSREG_UPDATE32

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_UPDATE64

Triggers when the system updates a register. Fields:

CRm unsigned int
: CRm

CRn unsigned int
: CRn

NS bool
: Secure or nonsecure banked register is accessed.

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

SYSREG_WRITE64

System Coprocessor register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CRm unsigned int
: CRm

CRn unsigned int
: CRn

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS bool
: Secure or nonsecure banked register is accessed.

REGNUM enum
: Internal register number

REGNUM_PHYS enum
: Internal register number physically accessed (e.g. in presense of VHE redirection)

UNDEF bool
: The register accessed is undefined.

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: The new value written.

opc unsigned int
: opcode 1

opc0 unsigned int
: opcode 0

opc2 unsigned int
: opcode 2

UNALIGNED_LDST_RETIRED

Processor unaligned load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1=>half-word, 2->word ...)

NUMBER_OF_BEATS unsigned int
: Number of accesses (beats) in this burst (Total size in bytes is ACCESS_SIZE * NUMBER_OF_BEATS)

VADDR unsigned int
: The virtual address of the access

VFP_D_REGS

VFP/NEON D 64 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_Q_REGS

VFP/NEON Q 128 bit register write. Fields:

ALIASING enum
: Type of register aliasing used

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MASK unsigned int
: Each bit indicates a write to the corresponding byte in the register. For example, 0x1 would indicate a write to VALUE[7:0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

VFP_SYS_REGS

Writes to the VFP/NEON units system registers. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID enum
: Which VFP system register is written.

MASK unsigned int
: The mask indicating updated cumulative bits.

OLD_VALUE unsigned int
: The register's old value overwritten.

VALUE unsigned int
: The new value written to the VFP system register.

VFP_S_REGS

VFP/NEON S 32 bit register write. Fields:

ALIASING enum
: Type of register aliasing used.

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number.

MASK unsigned int
: Each bit indicates a write to the corresponding bit in the register. For example, 0x1 would indicate a write to VALUE[0]

OLD_VALUE unsigned int
: The old value overwritten.

VALUE unsigned int
: The new value written to the register.

WAYPOINT

Signals end of basic block execution. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: Origin instructions set

IS_COND bool
: Indicates if this is a conditional waypoint.

PC unsigned int
: Origin address (or 0 if unavailable)

TAKEN bool
: Indicates if this waypoint was taken.

TARGET unsigned int
: Destination address

TARGET_ISET enum
: Destination instructions set

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

WFE_END

WFE ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFE.

WFE_EVENT_REGISTER

WFE event register status: set/clear, reason. Fields:

INST_COUNT unsigned int
: Ticks count.

REASON enum
: Reason for set/clear. Only REASON==0 (Cleared by WFE) clears the bit, all other reasons set it.

WFE_IGNORED

WFE ignored. Fields:

EVENT bool
: This WFE was ignored because the event register was set.

INST_COUNT unsigned int
: Ticks count when ignoring WFE.

TRAPPED bool
: This WFE was trapped.

WFE_START

WFE entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFE.

WFI_END

WFI ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFI.

WFI_IGNORED

WFI ignored. Fields:

DISABLED bool
: This WFI was ignored because WFI is disabled.

INST_COUNT unsigned int
: Ticks count when ignoring WFI.

REASON enum
: specifies reason why WFI trace was ignored

TRAPPED bool
: This WFI was trapped.

WFI_START

WFI entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFI.

WFI_WAKEUP

WFI wakeup. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT `unsigned int`
: Ticks count when WFI wakeup occurred.

REASON `enum`
: Reason for wakeup.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.46 ARM_SC000 - trace

This section describes the trace sources.

ASYNC_MEMORY_FAULT

Context ID Register write. Fields:

FAULT unsigned int
: Fault status

PADDR unsigned int
: Physical Address (or 0 if unavailable)

ATOMIC_END_ACCESS

Bus trace access for atomics. Fields:

ACCESS_FAIL bool
: Memory access failed

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

LOAD_VALUE unsigned int
: Loaded values

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ATOMIC_START_ACCESS

Bus trace access for atomics. Fields:

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ArchMsg.Error.exit_code

Fields:

EXITCODE signed int
: Exit code to be returned

KIND string
: Component kind that invoked the exit code trace

REASON string
: Reason why the exit code trace was invoked (optional)

ArchMsg.Warning.TarmacReconstructor

Unexpected event happened during Tarmac reconstruction. Fields:

REASON string
: Reason

ArchMsg.Warning.Unpredictably Indexed PM Event Register

Fields:

IsDirect unsigned int
: Direct Access rather than indirect

N unsigned int
: Number of Accessible Registers

SEL unsigned int
: Selector

ArchMsg.Warning.branch_to_unaligned_address

Fields:

ADDR unsigned int
: Unaligned branch address

ArchMsg.Warning.dap_csw_bad_size

A write to CM3DAP CSW has an invalid size field. Fields:

DATA unsigned int
: bits[2:0] are size

ArchMsg.Warning.dcimvac_matches_watchpoint

Watchpoints matching an AArch32 DCIMVA are implementation defined.

ArchMsg.Warning.decode_fieldmismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_fracbitsnegative

VCVT Instruction is unpredictable with a negative number of fraction bits.

ArchMsg.Warning.decode_initblock

Instruction is unpredictable in an IT block.

ArchMsg.Warning.decode_initblocknotlast

Instruction is unpredictable in an IT block when not the last.

ArchMsg.Warning.decode_invalid_condition

Instruction is unpredictable when the condition is not AL.

ArchMsg.Warning.decode_invalidfieldcombination

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_invalidvalue

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_mem_hint_unallocated

There is no memory hint allocated to this bit pattern.

ArchMsg.Warning.decode_registermatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registermismatch

Fields:

FIELD1 enum
: Field name

FIELD2 enum
: Field name

ArchMsg.Warning.decode_registeroutofrange

Float point instruction is unpredictable indexing beyond the end of the register bank.

ArchMsg.Warning.decode_sbzsbo

Reserved bits in the instruction are not canonical.

ArchMsg.Warning.decode_transactiontoobig

Attempt to perform a memory transaction of more than 128 bytes.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.decode_unpred_other

Instruction is unpredictable.

ArchMsg.Warning.decode_unpreduseofpc

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13

Fields:

FIELD enum
: Field name

ArchMsg.Warning.decode_unpreduseofr13inreglist

Instruction is unpredictable with R13 in register-list.

ArchMsg.Warning.decode_useofd16orhigher

Attempt to use register D16 or higher where it is not allowed.

ArchMsg.Warning.decode_writebackandbaseinlist

Instruction is unpredictable with write back when the base register is in the transfer list.

ArchMsg.Warning.decode_zeroregistersinlist

Instruction is unpredictable when no registers are to be transfered.

ArchMsg.Warning.recursive_branch

An instruction is performing a branch that targets the same instruction. If this was intended then a WFI might be more effective. Fields:

PC unsigned int
: Address of instruction causing the branch

ArchMsg.Warning.recursive_exception

An instruction has generated an exception that targets the same instruction. Fields:

CPSR unsigned int
: Processor state

PC unsigned int
: Address of instruction causing the exception

TYPE enum
: Type of exception

ArchMsg.Warning.reentrant_vector_catch

Fields:

ADDR unsigned int
: vector address

ArchMsg.Warning.reserved_it_state

Fields:

ITSTATE unsigned int
: reserved IT state

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.secure_vector_fetch_from_nonsecure

Secure vector table is being fetched from Non-secure memory .

ArchMsg.Warning.sg_in_it_block

Arm recommends that software does not place SG (security gate instruction) inside an IT instruction block.

ArchMsg.Warning.unpredictable_a32_breakpoint

Fields:

ADDR1 unsigned int
: Address

ADDR2 unsigned int
: Breakpoint address

BAS unsigned int
: BAS Field

IS_ADDRESS_MISMATCH_BREAKPOINT bool
: Is Mismatch breakpoint, not a match breakpoint

N unsigned int
: Breakpoint number

OFFSET unsigned int
: Indicates which halfword actually considering into the A32 instruction

ArchMsg.Warning.unpredictable_banked_register_access

Fields:

R unsigned int
: R bit

SYSM unsigned int
: SYSm

ArchMsg.Warning.unpredictable_ccfail_undef

Fields:

OPCODE unsigned int
: opcode of the instruction

VADDR unsigned int
: Virtual address of the instruction being executed

ArchMsg.Warning.unpredictable_ici_bits

ICI bits do not apply or are architecturally unpredictable for instruction.

ArchMsg.Warning.unpredictable_t32_breakpoint

Fields:

ADDR1 unsigned int
: Address

ADDR2 unsigned int
: Breakpoint address

BAS unsigned int
: BAS Field

IS_ADDRESS_MISMATCH_BREAKPOINT bool
: Is Mismatch breakpoint, not a match breakpoint

N unsigned int
: Breakpoint number

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_t32_vector_catch

Fields:

ADDR unsigned int
Address
OFFSET unsigned int
Indicates which halfword actually considering into the T32 instruction

ArchMsg.Warning.unpredictable_watchpoint_far

Fields:

LOWER_BOUND unsigned int
the lowest address accessed by the instruction that triggered the watchpoint
UPPER_BOUND unsigned int
the highest watchpointed address accessed by that instruction

ArchMsg.Warning.warning_AdvSIMDExpandImmUnexpectedZero

AdvSIMDExpandImm may treat this immediate value as UNPREDICTABLE.

ArchMsg.Warning.warning_ConditionalSMC

SMC instruction has UNPREDICTABLE effects when conditional when combined with traps.

ArchMsg.Warning.warning_access_crosses_page_boundary

Access crosses page boundary.

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_has_fault_on_both_pages

Access crosses page boundary has fault(s).

ArchMsg.Warning.warning_access_crosses_page_boundary_spanning_different_memory

Fields:

ADDR unsigned int
address of access crossing page
MEMTYPE_PAGE1 signed int
First Page's memory type (0-Normal, 1-Non-Normal)
MEMTYPE_PAGE2 signed int
Second Page's memory type (0-Normal, 1-Non-Normal)

ArchMsg.Warning.warning_access_wraps_around_memory

Fields:

WIDTH unsigned int
Address width

ArchMsg.Warning.warning_bx_from_thumbee

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ADDR unsigned int
Destination address

ArchMsg.Warning.warning_ccsidr_unimplemented_level

Fields:

CSSELR unsigned int
current effective value of CSSELR

ArchMsg.Warning.warning_change_to_ns_when_tge_set

Attempting to change NS to 1 when HCR.TGE = 1.

ArchMsg.Warning.warning_contiguous_bit_check_abort

Fields:

ENTRY_ADDR unsigned int
address of conflicting TLB entry

ArchMsg.Warning.warning_contiguous_bit_error

Fields:

CONTIG_ADDR unsigned int
address of the TLB first read, with which this entry is expected to be contiguous
CONTIG_DATA unsigned int
expected contents based on the entry at CONTIG_ADDR
ENTRY_ADDR unsigned int
address of conflicting TLB entry
ENTRY_DATA unsigned int
contents of conflicting TLB entry

ArchMsg.Warning.warning_cp10_cp11_mismatch

Fields:

EL unsigned int
EL owning the control

ArchMsg.Warning.warning_cp10_cp11_reserved_value

Fields:

CP10 unsigned int
CP10 access
CP11 unsigned int
CP11 access

ArchMsg.Warning.warning_csselr_level_out_of_range

Fields:

IMPLEMENTED unsigned int
number of cache levels implemented
LEVEL unsigned int
written value of CSSELR.Level
TYPE enum
The type of cache selected by CSSELR.InD and TnD

ArchMsg.Warning.warning_debug_flow_control_bits_not_obeyed

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ERROR enum
: type of error resulting

REGISTER enum
: register under consideration

ArchMsg.Warning.warning_debug_register_access_during_reset

Fields:

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

ArchMsg.Warning.warning_decode_cps_inconsistent_fields

The mode setting fields in the CPS instruction are inconsistent.

ArchMsg.Warning.warning_decode_invalid_state

Instruction is unpredictable in current exception-level/security state.

ArchMsg.Warning.warning_default_cacheable_mmu_on

Default cacheable is enabled (HCR.DC = 1) while MMU is enabled.

ArchMsg.Warning.warning_default_cacheable_vmmu_off

Default cacheable is enabled (HCR.DC = 1) while HCR.VM = 0.

ArchMsg.Warning.warning_deprecated_wvr_bit_2_set

Fields:

N unsigned int
: index

WVR unsigned int
: DBGWVR

ArchMsg.Warning.warning_eret_while_software_step_active_pending

Missing ISB between setting MDSCR_EL1.SS and ERET.

ArchMsg.Warning.warning_exclusive_to_non_normal

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_exclusive_to_non_writeback

Fields:

ADDR unsigned int
: Address

ArchMsg.Warning.warning_execute_from_device_memory

An attempt was made to execute from Device memory.

ArchMsg.Warning.warning_illegal_cpsr_mode

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NEW_MODE unsigned int
New value of CPSR.M

ArchMsg.Warning.warning_illegal_srs_mode

Fields:

MODE unsigned int
mode for Banked SP

ArchMsg.Warning.warning_implementation_defined_read_debug_register_in_SCS

Read Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

PPB_OFFSET unsigned int
debug register offset

ArchMsg.Warning.warning_implementation_defined_sequential_security_transitions_supported

The behaviour of a sequential instruction fetches that cross from non-secure to secure memory and contain SG instruction is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_implementation_defined_stack_limit_check_supported

It is IMPLEMENTATION DEFINED whether stack pointer limit checking is performed for the this instructions. Fields:

ADDRESS unsigned int
Address of instruction
OPCODE unsigned int
instruction opcode

ArchMsg.Warning.warning_implementation_defined_write_debug_register_in_SCS

Write Access to Debug registers from the processor is IMPLEMENTATION DEFINED. Fields:

DATA unsigned int
data attempted to be written
PPB_OFFSET unsigned int
debug register offset

ArchMsg.Warning.warning_invalid_tcr_granule

Fields:

REQUEST signed int
page size requested (or 0 for reserved)
SUBSTITUTE signed int
best guess available page size
TG_ID bool
bits TG0 or TG1

ArchMsg.Warning.warning_load_multiple_user_registers_from_user_mode

An LDM(user registers) instruction executed from user mode.

ArchMsg.Warning.warning_load_pc_from_unaligned

PC loaded from an unaligned location.

ArchMsg.Warning.warning_mte_in_mem_attr_not_supported

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ATTR unsigned int
Memory attribute

ArchMsg.Warning.warning_reserved_breakpoint_state_match

Fields:

IS_BREAKPOINT unsigned int
Is breakpoint, not watchpoint

ArchMsg.Warning.warning_shareability

Unpredictable: combination of the 1st and 2nd stages of translation is Normal Inner Non-Cacheable, Outer Non-Cacheable.

ArchMsg.Warning.warning_software_step_set_while_enabled

MDSCR_EL1.SS set to 1 while software step debug exceptions are enabled.

ArchMsg.Warning.warning_thumb_instruction_wraps_around_memory

Thumb instruction warps around memory.

ArchMsg.Warning.warning_ttbr_sbz_bits_are_not_zero

Fields:

MSB unsigned int
x-1
TTBR enum
which TTBR register

ArchMsg.Warning.warning_unaligned_address_dbgdtrrx_write

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_address_dbgdtrtx_read

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_device

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unaligned_to_strongly_ordered

Fields:

ADDR unsigned int
Address

ArchMsg.Warning.warning_unknown_sau_rnr

SAU_RNR was set to an unsupported value.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_AIRCR_PRIP_and_BFHFNMINP

The effect of setting both AIRCR.BFHFNMINP and AIRCR.PRIP to 1 is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTCLRACTIVE_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTCLRACTIVE if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_and_SYSRESETREQ

When the processor is halted in Debug state, if a write to the register writes a 1 to both VECTRESET and SYSRESETREQ, the behavior is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_VECTRESET_when_not_in_debug

The effect of writing a 1 to AIRCR.VECTRESET if the processor is not halted in Debug state is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_AIRCR_incorrect_VKEY

The value 0x05FA must be written to AIRCR.VKEY, otherwise the register write is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_EXE_RETURN_Reserved_Bit

EXC_RETURN[23:7] are reserved with the special condition that all bits should be written as one. Values other than all 1s are UNPREDICTABLE. Fields:

EXC_RETURN `unsigned int`
: Special PC value

SBO_MASK `unsigned int`
: Which bits Should Be One to not UNPRED

ArchMsg.Warning.warning_unpredictable_change_to_DEMCR_MON_STEP_at_insufficient_priority

The effect of changing DEMCR.MON_STEP at an execution priority that is lower than the priority of the DebugMonitor exception is CONSTRAINED UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_change_to_priority_of_active_exception

Changing the priority of an active exception is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_clear_lspact

Fields:

DRVEXC_TAKEN `unsigned int`
: derived exception taken not pended

REACHED_LAST_STORE `unsigned int`
: only the last faulted

ArchMsg.Warning.warning_unpredictable_dbg_exit_unaligned_dlr

Debug exit to AArch32 with DLR[0] set to 1. PC[0] can be set to 0 or to DLR[0].

ArchMsg.Warning.warning_unpredictable_exception_catch

Generation of an exception catch event is unpredictable.

ArchMsg.Warning.warning_unpredictable_exception_return_inconsistent_state

state on exception return is unpredictable.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_exception_return_instruction

exception return instruction is unpredictable.

ArchMsg.Warning.warning_unpredictable_in_debug_state

Instruction is UNPREDICTABLE when executed in debug state.

ArchMsg.Warning.warning_unpredictable_mair_encoding

Fields:

ATTR unsigned int
: Memory attribute

ArchMsg.Warning.warning_unpredictable_pmu_counter_access

Access to PMU counters from non-secure EL0 or EL1 is UNPREDICTABLE with MDCR_EL2.HPMN set to 0.

ArchMsg.Warning.warning_unpredictable_prioritization_breakpoint_match_vector_catch

Two events with the same priority occurred at the same instruction: Address Matching Vector Catch debug event, and Breakpoint debug event It is constrained unpredictable which is taken.

ArchMsg.Warning.warning_unpredictable_stack_selection

UNPREDICTABLE state on context switch.

ArchMsg.Warning.warning_unpredictable_tsize_out_of_range

Fields:

TCR enum
: Which TCR

TSIZE signed int
: Value of TSize

TSIZE_MAX signed int
: The maximum allowed value

TTBR unsigned int
: Which TTBR region

ArchMsg.Warning.warning_unpredictable_unaligned_pc_as_base_register

The PC value must be word-aligned, otherwise the behavior of the instruction is UNPREDICTABLE.

ArchMsg.Warning.warning_unpredictable_unaligned_pop_stack

Fields:

FRAME_ADDR unsigned int
: unaligned stack frame address

ArchMsg.Warning.warning_unpredictable_vmsa_memattrib

Fields:

TEXCB unsigned int
: texcb value

TEX_REMAP bool
: use tex remap

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.warning_unpredictable_vtcr_t0sz_sl0

Fields:

SL0 signed int
: VTCR.SL0, starting level for stage 2 translation table walks

T0SZ signed int
: VTCR.T0SZ, required input address range

ArchMsg.Warning.warning_unpredictable_write_DHCSR

Fields:

NEW unsigned int
: data attempted to be written

OLD unsigned int
: previous value

ArchMsg.Warning.warning_unpredictable_write_SHCSR

UNPREDICTABLE setting a pending bit to 1 for an exception with priority >= execution priority.

ArchMsg.Warning.warning_unsupported_access_to_memory_mapped_register

Access to memory mapped register is unsupported.

ArchMsg.Warning.warning_user_jmcr_access

Fields:

WRITE unsigned int
: access is write

ArchMsg.Warning.warning_wcr_mask_and_bas

DBGWCR_EL1.MASK is non-zero and BAS is not 0xff.

ArchMsg.Warning.warning_wcr_mask_reserved

DBGWCR_ELn.MASK is set to a reserved value.

ArchMsg.Warning.warning_wcr_non_configuous_bas

DBGWCR_EL1.BAS has non-contiguous set of ones.

ArchMsg.Warning.warning_wvr_masked_bit_not_zero

DBGWVRn_EL1.VA contains a bit masked by DBGWCRn_EL1.MASK that is not zero.

ArchMsg.warning_unpred_system_register_access

Fields:

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

BRANCH_MISPREDICT

Simulating branch mispredict. Fields:

PC unsigned int
: Origin address (or 0 if unavailable)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

BRA_DIR

Direct branches, to immediate address. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR

Indirect branches, perhaps to a register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR_CCFAIL

Indirect branches fail, perhaps to a register.

BRA_RET_CCFAIL

Branche return fail, perhaps to a register.

CCFAIL

Conditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CCFAIL_UNC

Unconditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS

Conditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS_UNC

Unconditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CODE_CACHE_MAINT

Code cache maintenance. Fields:

END_INCL unsigned int
: Inclusive end addres for by VA or PA requests

NS bool
: Non-secure world for PA

START unsigned int
: Start address for by VA or PA requests

TYPE enum
: Request type.

COMPILE_BLOCK_END

Last instruction of basic block translated. Fields:

VADDR unsigned int
: Address of next instruction after this basic block

COMPILE_BLOCK_START

First instruction of basic block translated. Fields:

VADDR unsigned int
: Address of first instruction in basic block

COMPILE_INST

ARM instruction compiled. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DISASS string
: Disassembly of the instruction.

ISET enum
: The instruction set of this instruction.

ITSTATE unsigned int
: The ITSTATE current for the instruction.

OPCODE unsigned int
: The opcode of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

CONTEXTIDR

Context ID Register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

NS bool
: Secure or nonsecure banked register is accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

CONTEXT_SYNC

Called for every context synchronization event. Fields:

ARCH_CSE bool
: Architectural Context Synchronization Event

SYSREG_SYNC bool
: System register synchronization event

CORE_ENDIAN

Core BE8 Big-Endian state changed. Fields:

BE8 bool
: Core BE8 Big-Endian state

CORE_INFO

Static processor attributes. Only triggered by a call to DumpState(). Fields:

ARCH_PROFILE enum
: The architecture profile of the core.

CLUSTER_ID unsigned int
: The cluster ID of this processor.

CORE_NUM unsigned int
: The number of this core in an MP processor.

DETAILED_CORE_FEATURES enum
: Detailed core features as a bitfield. Enum values are bitmasks to extract the info from the field

FPU_VERSION enum
: The VFP version implemented by the core.

MEM_ARCH enum
: The memory architecture of the core.

MIDR_EL1 unsigned int
: MIDR_EL1 value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NUM_CORES unsigned int
: The number of cores in this MP processor.

QUANTUM_SIZE unsigned int
: The default quantum size of the core.

SECURITY_FEATURES bool
: Does the core have security features?

CORE_LOADS

Processor load accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CORE_REGS

Changes of the core registers R0 to R14. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ID unsigned int
: The register number, 0 to 14.

OLD_VALUE unsigned int
: The old value overwritten.

PHYS_ID enum
: The physical register accessed.

VALUE unsigned int
: The new value written to the register.

CORE_STORES

Processor store accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CRYPTO_SPEC

Every crypto instruction speculatively executed.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DATA_CACHE_SET_ALLOC_TAGS

Set allocation tags from DC GVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of memory in bytes.

TAG unsigned int
: The value of the tag being set.

VADDR unsigned int
: Virtual address used for setting allocation tags.

DATA_CACHE_ZERO

Zero and invalidate cache line from DC ZVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of zero'd memory in bytes.

VADDR unsigned int
: Virtual address of the zero'd cache line.

DEBUG_EVENT

Hardware debug support event. Fields:

EVENT enum
: Description of event

VALUE unsigned int
: data value

DWT_MATCH

DWT comparator matches. Fields:

NUM unsigned int
: DWT comparator number

TYPE enum
: DWT comparator configuration

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

END_COMPILE

Compilation end. Fields:

END_OF_PAGE_COUNT unsigned int
: Number of basic blocks exited due to page boundary since START_COMPILE.

FETCHFAIL_COUNT unsigned int
: Number of basic blocks exited due to ifetch failure START_COMPILE.

INST_COUNT unsigned int
: Number of instructions compiled since START_COMPILE.

NONSEQ_COUNT unsigned int
: Number of basic blocks exited due to non-sequential instructions since START_COMPILE.

PAGE_STRADDLE_COUNT unsigned int
: Number of basic blocks exited due to unaligned instructions crossing page since START_COMPILE.

EXCEPTION

Exceptions which are taken. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: The instruction set of the processor when the exception occurred.

LR unsigned int
: The value assigned to the link register.

PC unsigned int
: The location where the exception occurred.

TARGET_ISET enum
: The instruction set of the exception handler code.

TARGET_PC unsigned int
: The address the exception branches to.

VECTOR enum
: The exception vector.

EXCEPTION_END

Every exception completed.

EXCEPTION_ENTRY

Event marking the entry of an exception. All pushes of register to the stack will follow this event. Fields:

PC unsigned int
: The location where the exception occurred.

VECTOR enum
: The exception that occurred.

EXCEPTION_RAISE

Every exception raised.

EXCEPTION_RETURN

Marks the end of an exception, but on an M core does not indicate a program flow change (branch). In most situations it will be followed by an indirect branch. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ISET enum
: The instructions set of the branch instruction.

PC unsigned int
: The address of the branch instruction.

EXCEPTION_RETURN_PREBRANCH

Exception return event that must occur before PC and processor state have updated.

EXCEPTION_START

Every exception started.

EXCEPTION_VECTOR_FETCH

I-side vector fetch for M-class exception. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Result of fetch if OK

RESPONSE enum
: 0=Aborted, 1=OK

VADDR unsigned int
: Where the vector is fetched from (including VTOR)

FP_STATE

Floating point state. Fields:

FPSR unsigned int
: FPSR register value.

MASK unsigned int
: The mask indicating updated cumulative bits.

HLT

HLT instruction occurred. Fields:

IMM16 unsigned int
: The 16-bit immediate value encoded in HLT instruction.

INFO_EXCEPTION_REASON

Provide information of real cause of exception. Fields:

FaultCause enum
: Which bit of which V7M/V8M-mainline FSR/HFSR would have been set.

PC unsigned int
: The location where the exception occurred.

PHASE enum
: What we are doing with the exception.

REASONS enum
: Why we are doing it.

VECTOR enum
: The exception that occurred.

INFO_STACKING

Provide information of real cause of exception. Fields:

PHASE enum
: Are we starting or ending a sequence of stack writes.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

REASONS enum
: Why we are doing it.

INST

Every instruction executed. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DEBUG_STATE bool
: The instruction is executed in debug state.

DISASS string
: Disassembly of instruction.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The current instruction set.

ITSTATE unsigned int
: The current ITSTATE.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

MODE enum
: The mode the core is in.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

OPCODE unsigned int
: The opcode of the instruction.

PADDR unsigned int
: The physical address of the instruction.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

INST_END

Every instruction completed.

INST_START

Every instruction started. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

INST_STRADDLE

Instruction straddles boundary. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

ISB

ISB instruction occurred.

LOAD_MULTIPLE_REGS_LIST

List of destination registers (not source) of a multiple registers load instruction. Fields:

REG string
: Register to add into the list

LOCAL_MONITOR

Local monitor activity. Fields:

PADDR unsigned int
: Local Monitor Address

State enum
: State of the monitor (Open/Exclusive)

LOCKUP_CYCLE

This event is triggered if the core enters or stays in the lock-up state. Fields:

CAUSE enum
: What causes the lockup?.

LOCKUP_ENTRY

This event is triggered if the core enters the lock-up state. Fields:

PC unsigned int
: The PC when the lockup state is entered.

LOCKUP_EXIT

This event is triggered if the core leaves the lock-up state.

MODE_CHANGE

Mode change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: The new mode.

OLD_MODE enum
: The old mode.

MPU_TRANS

Address translation information. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

EXEC_PERM enum
: Execution Permission

INNERCACHE_RA bool
: Is the inner cache allocate on read

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

MEMTYPE enum
: Memory type.

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

PADDR unsigned int
: Address of the access.

PAGESIZE unsigned int
: Page size as log2(size).

READ_PERM enum
: Read Permission

REG_NUM signed int
: MPU region-number 0-15 (else 0xFF=none, 0xBB/0xDD=background/default)

SH enum
: Shareability

SIDE enum
: Inst / Data.

WRITE_PERM enum
: Write Permission

PERIODIC

Called for every quantum. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

PC unsigned int
: The address of the next instruction to be executed on this CPU.

PPB_READ_ACCESS

Trace reads from the core to memory mapped registers in the PPB (private peripheral bus) address range. Fields:

ACCESS_SIZE unsigned int
: Logarithm of the access size: 0=byte, 1=halfword, 2=word, 3=doubleword, ...

BURST_LEN unsigned int
: The number of data transfers (beats) in this burst.

DATA unsigned int
: The data transferred.

INNER_ATTRIBUTES unsigned int
: The inner attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

INST bool
: Is an instruction fetch.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOCK enum
: Normal / Locked / Exclusive.

MASTER_ID unsigned int
: The AXI master ID.

NS bool
: Is a nonsecure access.

OUTER_ATTRIBUTES unsigned int
: The outer attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

PADDR unsigned int
: Physical address of access.

PRIV bool
: Is privileged access.

RESPONSE enum
: Whether the transaction was successful, or an error occurred.

USER_FLAGS unsigned int
: Core specific additional signals.

PPB_WRITE_ACCESS

Trace writes from the core to memory mapped registers in the PPB (private peripheral bus) address range. Fields:

ACCESS_SIZE unsigned int
: Logarithm of the access size: 0=byte, 1=halfword, 2=word, 3=doubleword, ...

BURST_LEN unsigned int
: The number of data transfers (beats) in this burst.

DATA unsigned int
: The data transferred.

INNER_ATTRIBUTES unsigned int
: The inner attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

INST bool
: Is an instruction fetch.

LOCK enum
: Normal / Locked / Exclusive.

MASTER_ID unsigned int
: The AXI master ID.

NS bool
: Is a nonsecure access.

OUTER_ATTRIBUTES unsigned int
: The outer attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

PADDR unsigned int
: Physical address of access.

PRIV bool
: Is privileged access.

RESPONSE enum
: Whether the transaction was successful, or an error occurred.

USER_FLAGS unsigned int
: Core specific additional signals.

PRELOAD_DATA

Data preload from PLD instruction. Fields:

VADDR unsigned int
: Virtual address of the data that should be preloaded.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PRELOAD_INST

Instruction preload from PLI instruction. Fields:

VADDR unsigned int
: Virtual address of the instruction that should be preloaded.

PRE_CORE_LOAD

Trace just before a core load. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_CORE_STORE

Trace just before a core store. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PSTATE_PACTIVE_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

PSTATE_PACTIVE_OPMODE_TRANSITION_DENIED

Trace the denied pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_POWER_TRANSITION

Trace pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

PSTATE_PACTIVE_POWER_TRANSITION_DENIED

Trace the denied pactive and power mode transitions on PChannel. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_THREAD_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions for threads on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

RUN_STATE

Run state transition. Fields:

INST_COUNT unsigned int
: Ticks count at point of transition.

NEW enum
: New run state.

OLD enum
: Old run state.

SEMIHOSTING_CALL

Call of a semihost function occurred. Fields:

PC unsigned int
: The program counter after the semihosting call.

SEMIHOSTING_PRECALL

About to call semihost. Fields:

EL2 bool
: Use EL2 translation regime.

NS bool
: Is Non-Secure.

REG_WIDTH unsigned int
: The current register width in bytes.

SIGNAL

External signal state change. Fields:

SIGNAL enum
: Signal that changed

STATE bool
: Signal asserted state

START_COMPILE

Compilation started. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VADDR unsigned int
: Instruction where compilation begins.

STG_MULTIPLE_MICRO_OPS_ADDRESS_RANGE

Address range for MTE stores with multiple uops. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

STORE_MULTIPLE_REGS_ADDRESS_RANGE

Address range for stores with multiple registers. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

SUBOPTIMAL_LDST_RETIRED

suboptimal load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1: half-word, 2: word ...)

OPTIMAL_ALIGN_SELECTOR unsigned int
: Selector to decide the optimal alignment size (1: access width, 2: 4-bytes, 3: 8-bytes, ..., 12: 4-kbytes)

VADDR unsigned int
: The virtual address of the access

SYNC

Called for every synchronization, that is every quantum break. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

LOCAL_QUANTUM unsigned int
: The local quantum of this CPU.

LOCAL_TIME unsigned int
: The local time of this CPU.

SYNC_REASON enum
: Reason for sync.

SYSCALL

System call instruction executed. Fields:

IMM unsigned int
: Immediate value of the system call instruction.

TYPE enum
: System call type.

VADDR unsigned int
: Instruction that caused the system call.

SYSREG_READ

Trace reads from system registers. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

REG_NAME enum
: Register number

VALUE unsigned int
: The value read.

SYSREG_UPDATE32

Triggers when the system updates a register. Fields:

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

SYSREG_UPDATE64

Triggers when the system updates a register. Fields:

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

SYSREG_WRITE

Trace write to system registers. Fields:

REG_ACCESS enum
: Register access status

REG_NAME enum
: Register number

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

VALUE unsigned int
: Value written to the register.

UNALIGNED_LDST_RETIRED

Processor unaligned load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1=>half-word, 2->word ...)

NUMBER_OF_BEATS unsigned int
: Number of accesses (beats) in this burst (Total size in bytes is ACCESS_SIZE * NUMBER_OF_BEATS)

VADDR unsigned int
: The virtual address of the access

WAYPOINT

Signals end of basic block execution. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: Origin instructions set

IS_COND bool
: Indicates if this is a conditional waypoint.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PC unsigned int
: Origin address (or 0 if unavailable)

TAKEN bool
: Indicates if this waypoint was taken.

TARGET unsigned int
: Destination address

TARGET_ISET enum
: Destination instructions set

WFE_END

WFE ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFE.

WFE_EVENT_REGISTER

WFE event register status: set/clear, reason. Fields:

INST_COUNT unsigned int
: Ticks count.

REASON enum
: Reason for set/clear. Only REASON==0 (Cleared by WFE) clears the bit, all other reasons set it.

WFE_IGNORED

WFE ignored. Fields:

EVENT bool
: This WFE was ignored because the event register was set.

INST_COUNT unsigned int
: Ticks count when ignoring WFE.

TRAPPED bool
: This WFE was trapped.

WFE_START

WFE entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFE.

WFI_END

WFI ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFI.

WFI_IGNORED

WFI ignored. Fields:

DISABLED bool
: This WFI was ignored because WFI is disabled.

INST_COUNT unsigned int
: Ticks count when ignoring WFI.

REASON enum
: specifies reason why WFI trace was ignored

TRAPPED bool
: This WFI was trapped.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

WFI_START

WFI entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFI.

WFI_WAKEUP

WFI wakeup. Fields:

INST_COUNT unsigned int
: Ticks count when WFI wakeup occurred.

REASON enum
: Reason for wakeup.

XPSR

Changes to the xPSR register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

OLD_VALUE unsigned int
: The old xPSR value

UNKNOWN unsigned int
: Bits within the register that have unknown value.

VALUE unsigned int
: The new xPSR value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.47 ARM_SC300 - trace

This section describes the trace sources.

ASYNC_MEMORY_FAULT

Context ID Register write. Fields:

FAULT unsigned int
: Fault status

PADDR unsigned int
: Physical Address (or 0 if unavailable)

ATOMIC_END_ACCESS

Bus trace access for atomics. Fields:

ACCESS_FAIL bool
: Memory access failed

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

LOAD_VALUE unsigned int
: Loaded values

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ATOMIC_START_ACCESS

Bus trace access for atomics. Fields:

ADDR unsigned int
: The virtual address of the access.

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

COMPARE_VALUE unsigned int
: Compare value for CAS

NSDESC unsigned int
: The security state of the access.

OPERAND_VALUE unsigned int
: Operation's operand

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OPERATION enum
: Operation type

PADDR unsigned int
: The physical address of the access.

PRIV bool
: Is this a privileged access?

ArchMsg.Error.exit_code

Fields:

EXITCODE signed int
: Exit code to be returned

KIND string
: Component kind that invoked the exit code trace

REASON string
: Reason why the exit code trace was invoked (optional)

ArchMsg.Warning.TarmacReconstructor

Unexpected event happened during Tarmac reconstruction. Fields:

REASON string
: Reason

ArchMsg.Warning.dap_csw_bad_size

A write to CM3DAP CSW has an invalid size field. Fields:

DATA unsigned int
: bits[2:0] are size

ArchMsg.Warning.recursive_branch

An instruction is performing a branch that targets the same instruction. If this was intended then a WFI might be more effective. Fields:

PC unsigned int
: Address of instruction causing the branch

ArchMsg.Warning.recursive_exception

An instruction has generated an exception that targets the same instruction. Fields:

CPSR unsigned int
: Processor state

PC unsigned int
: Address of instruction causing the exception

TYPE enum
: Type of exception

ArchMsg.Warning.secure_vector_fetch_from_nonsecure

Secure vector table is being fetched from Non-secure memory .

ArchMsg.Warning.sg_in_it_block

Arm recommends that software does not place SG (security gate instruction) inside an IT instruction block.

ArchMsg.warning_unpred_system_register_access

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

IS_WRITE unsigned int
: Write Not Read

OFFSET unsigned int
: Register Offset

BRANCH_MISPREDICT

Simulating branch mispredict. Fields:

PC unsigned int
: Origin address (or 0 if unavailable)

BRA_DIR

Direct branches, to immediate address. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR

Indirect branches, perhaps to a register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

IS_COND bool
: Indicates if this is a conditional branch.

IS_LINK bool
: Indicates if this is a branch with link.

PC unsigned int
: The address of the branch instruction.

TARGET_ISET enum
: The instructions set after the branch.

TARGET_PC unsigned int
: The address the instruction branches to.

BRA_INDIR_CCFAIL

Indirect branches fail, perhaps to a register.

BRA_RET_CCFAIL

Branche return fail, perhaps to a register.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CCFAIL

Conditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCFAIL_UNC

Unconditional instruction condition check fail. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS

Conditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CCPASS_UNC

Unconditional instruction condition check pass. Fields:

COND enum
: The condition of the conditional instruction.

CORE_NUM unsigned int
: Core number in a multi processor.

PC unsigned int
: The address of the conditional instruction.

CODE_CACHE_MAINT

Code cache maintenance. Fields:

END_INCL unsigned int
: Inclusive end addres for by VA or PA requests

NS bool
: Non-secure world for PA

START unsigned int
: Start address for by VA or PA requests

TYPE enum
: Request type.

COMPILE_BLOCK_END

Last instruction of basic block translated. Fields:

VADDR unsigned int
: Address of next instruction after this basic block

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

COMPILE_BLOCK_START

First instruction of basic block translated. Fields:

VADDR unsigned int
: Address of first instruction in basic block

COMPILE_INST

ARM instruction compiled. Fields:

DISASS string
: Disassembly of the instruction.

ISET enum
: The instruction set of this instruction.

ITSTATE unsigned int
: The ITSTATE current for the instruction.

OPCODE unsigned int
: The opcode of the instruction.

PC unsigned int
: The address of the instruction.

SIZE unsigned int
: The size of the instruction in bytes.

CONTEXTIDR

Context ID Register write. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

NS bool
: Secure or nonsecure banked register is accessed.

UNDEF bool
: The register accessed is undefined.

VALUE unsigned int
: The new value written.

CONTEXT_SYNC

Called for every context synchronization event. Fields:

ARCH_CSE bool
: Architectural Context Synchronization Event

SYSREG_SYNC bool
: System register synchronization event

CORE_ENDIAN

Core BE8 Big-Endian state changed. Fields:

BE8 bool
: Core BE8 Big-Endian state

CORE_INFO

Static processor attributes. Only triggered by a call to DumpState(). Fields:

ARCH_PROFILE enum
: The architecture profile of the core.

CLUSTER_ID unsigned int
: The cluster ID of this processor.

CORE_NUM unsigned int
: The number of this core in an MP processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DETAILED_CORE_FEATURES enum
: Detailed core features as a bitfield. Enum values are bitmasks to extract the info from the field

FPU_VERSION enum
: The VFP version implemented by the core.

MEM_ARCH enum
: The memory architecture of the core.

MIDR_EL1 unsigned int
: MIDR_EL1 value

NUM_CORES unsigned int
: The number of cores in this MP processor.

QUANTUM_SIZE unsigned int
: The default quantum size of the core.

SECURITY_FEATURES bool
: Does the core have security features?

CORE_LOADS

Processor load accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

TRANS bool
: Is this a translated access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VADDR unsigned int
: The virtual address of the access.

CORE_REGS

Changes of the core registers R0 to R14. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ID unsigned int
: The register number, 0 to 14.

OLD_VALUE unsigned int
: The old value overwritten.

PHYS_ID enum
: The physical register accessed.

VALUE unsigned int
: The new value written to the register.

CORE_STORES

Processor store accesses. Fields:

ACCESS_TYPE enum
: The type of instruction performing the access.

ACQREL enum
: Is this an acquire/release

ATTR unsigned int
: Memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

ATTR2 unsigned int
: Second page memory attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

DATA unsigned int
: The data read or written.

ELEMENT_SIZE unsigned int
: Width of each element.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

LOCK enum
: Normal, exclusive or locked access.

NSDESC unsigned int
: The physical address non-secure bit.

NSDESC2 unsigned int
: The second page physical address non-secure bit.

PADDR unsigned int
: The physical (translated) address.

PADDR2 unsigned int
: If different from PADDR, the physical address of the second page of the access.

RESPONSE enum
: 0=Aborted, 1=OK, 2=Exclusive Failed, 3=Unsuccessful

SIZE unsigned int
: Width of the access in bytes. Only required if DATA is not traced.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

TRANS bool
: Is this a translated access.

VADDR unsigned int
: The virtual address of the access.

CRYPTO_SPEC

Every crypto instruction speculatively executed.

DATA_CACHE_SET_ALLOC_TAGS

Set allocation tags from DC GVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of memory in bytes.

TAG unsigned int
: The value of the tag being set.

VADDR unsigned int
: Virtual address used for setting allocation tags.

DATA_CACHE_ZERO

Zero and invalidate cache line from DC ZVA instruction. Fields:

ATTR unsigned int
: Transaction Attributes: [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

NSDESC unsigned int
: The security state of the access.

PADDR unsigned int
: The physical address of the access.

RESPONSE enum
: 0=Aborted, 1=OK

SIZE unsigned int
: The size of zero'd memory in bytes.

VADDR unsigned int
: Virtual address of the zero'd cache line.

DEBUG_EVENT

Hardware debug support event. Fields:

EVENT enum
: Description of event

VALUE unsigned int
: data value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DWT_MATCH

DWT comparator matches. Fields:

NUM unsigned int
: DWT comparator number

TYPE enum
: DWT comparator configuration

END_COMPILE

Compilation end. Fields:

END_OF_PAGE_COUNT unsigned int
: Number of basic blocks exited due to page boundary since START_COMPILE.

FETCHFAIL_COUNT unsigned int
: Number of basic blocks exited due to ifetch failure START_COMPILE.

INST_COUNT unsigned int
: Number of instructions compiled since START_COMPILE.

NONSEQ_COUNT unsigned int
: Number of basic blocks exited due to non-sequential instructions since START_COMPILE.

PAGE_STRADDLE_COUNT unsigned int
: Number of basic blocks exited due to unaligned instructions crossing page since START_COMPILE.

EXCEPTION

Exceptions which are taken. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: The instruction set of the processor when the exception occurred.

LR unsigned int
: The value assigned to the link register.

PC unsigned int
: The location where the exception occurred.

TARGET_ISET enum
: The instruction set of the exception handler code.

TARGET_PC unsigned int
: The address the exception branches to.

VECTOR enum
: The exception vector.

EXCEPTION_END

Every exception completed.

EXCEPTION_ENTRY

Event marking the entry of an exception. All pushes of register to the stack will follow this event. Fields:

PC unsigned int
: The location where the exception occurred.

VECTOR enum
: The exception that occurred.

EXCEPTION_RAISE

Every exception raised.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

EXCEPTION_RETURN

Marks the end of an exception, but on an M core does not indicate a program flow change (branch). In most situations it will be followed by an indirect branch. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The core's instruction counter, starting at 1 for the first instruction.

ISET enum
: The instructions set of the branch instruction.

PC unsigned int
: The address of the branch instruction.

EXCEPTION_RETURN_PREBRANCH

Exception return event that must occur before PC and processor state have updated.

EXCEPTION_START

Every exception started.

EXCEPTION_VECTOR_FETCH

I-side vector fetch for M-class exception. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

DATA unsigned int
: Result of fetch if OK

RESPONSE enum
: 0=Aborted, 1=OK

VADDR unsigned int
: Where the vector is fetched from (including VTOR)

FP_STATE

Floating point state. Fields:

FPSR unsigned int
: FPSR register value.

MASK unsigned int
: The mask indicating updated cumulative bits.

HLT

HLT instruction occurred. Fields:

IMM16 unsigned int
: The 16-bit immediate value encoded in HLT instruction.

INFO_EXCEPTION_REASON

Provide information of real cause of exception. Fields:

FaultCause enum
: Which bit of which V7M/V8M-mainline FSR/HFSR would have been set.

PC unsigned int
: The location where the exception occurred.

PHASE enum
: What we are doing with the exception.

REASONS enum
: Why we are doing it.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VECTOR enum
The exception that occurred.

INFO_STACKING

Provide information of real cause of exception. Fields:

PHASE enum
Are we starting or ending a sequence of stack writes.
REASONS enum
Why we are doing it.

INST

Every instruction executed. Fields:

CORE_NUM unsigned int
Core number in a multi processor.
CURRENT_TIME unsigned int
The core's current time, as simulated time plus local time.
DEBUG_STATE bool
The instruction is executed in debug state.
DISASS string
Disassembly of instruction.
INST_COUNT unsigned int
The core's instruction counter, starting at 1 for the first instruction.
ISET enum
The current instruction set.
ITSTATE unsigned int
The current ITSTATE.
LOCAL_TIME unsigned int
The core's local time, relative to the current quantum.
MODE enum
The mode the core is in.
NSDESC unsigned int
The physical address non-secure bit.
NSDESC2 unsigned int
The second page physical address non-secure bit.
OPCODE unsigned int
The opcode of the instruction.
PADDR unsigned int
The physical address of the instruction.
PADDR2 unsigned int
If different from PADDR, the physical address of the second page of the instruction.
PC unsigned int
The address of the instruction.
SIZE unsigned int
The size of the instruction in bytes.

INST_END

Every instruction completed.

INST_START

Every instruction started. Fields:

ISET enum
The current instruction set.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

INST_STRADDLE

Instruction straddles boundary. Fields:

ISET enum
: The current instruction set.

MODE enum
: The mode the core is in.

NS enum
: The current Secure State.

PC unsigned int
: The address of the conditional instruction.

ISB

ISB instruction occurred.

ITM

Instrumentation Trace Macrocell. Fields:

ITM_PACKET_TYPE enum
: ITM and DWT packets type.

PACKET_HEADER unsigned int
: ITM Packet Header.

PACKET_PAYLOAD unsigned int
: ITM Packet Payload.

LOAD_MULTIPLE_REGS_LIST

List of destination registers (not source) of a multiple registers load instruction. Fields:

REG string
: Register to add into the list

LOCAL_MONITOR

Local monitor activity. Fields:

PADDR unsigned int
: Local Monitor Address

State enum
: State of the monitor (Open/Exclusive)

LOCKUP_CYCLE

This event is triggered if the core enters or stays in the lock-up state. Fields:

CAUSE enum
: What causes the lockup?.

LOCKUP_ENTRY

This event is triggered if the core enters the lock-up state. Fields:

PC unsigned int
: The PC when the lockup state is entered.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LOCKUP_EXIT

This event is triggered if the core leaves the lock-up state.

MODE_CHANGE

Mode change. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

MODE enum
: The new mode.

OLD_MODE enum
: The old mode.

MPU_TRANS

Address translation information. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

EXEC_PERM enum
: Execution Permission

INNERCACHE_RA bool
: Is the inner cache allocate on read

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

MEMTYPE enum
: Memory type.

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

PADDR unsigned int
: Address of the access.

PAGESIZE unsigned int
: Page size as log2(size).

READ_PERM enum
: Read Permission

REG_NUM signed int
: MPU region-number 0-15 (else 0xFF=none, 0xBB/0xDD=background/default)

SH enum
: Shareability

SIDE enum
: Inst / Data.

WRITE_PERM enum
: Write Permission

PERIODIC

Called for every quantum. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

INST_COUNT unsigned int
: The instruction count of this CPU.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PC unsigned int
: The address of the next instruction to be executed on this CPU.

PPB_READ_ACCESS

Trace reads from the core to memory mapped registers in the PPB (private peripheral bus) address range. Fields:

ACCESS_SIZE unsigned int
: Logarithm of the access size: 0=byte, 1=halfword, 2=word, 3=doubleword, ...

BURST_LEN unsigned int
: The number of data transfers (beats) in this burst.

DATA unsigned int
: The data transferred.

INNER_ATTRIBUTES unsigned int
: The inner attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

INST bool
: Is an instruction fetch.

LOCK enum
: Normal / Locked / Exclusive.

MASTER_ID unsigned int
: The AXI master ID.

NS bool
: Is a nonsecure access.

OUTER_ATTRIBUTES unsigned int
: The outer attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

PADDR unsigned int
: Physical address of access.

PRIV bool
: Is privileged access.

RESPONSE enum
: Whether the transaction was successful, or an error occurred.

USER_FLAGS unsigned int
: Core specific additional signals.

PPB_WRITE_ACCESS

Trace writes from the core to memory mapped registers in the PPB (private peripheral bus) address range. Fields:

ACCESS_SIZE unsigned int
: Logarithm of the access size: 0=byte, 1=halfword, 2=word, 3=doubleword, ...

BURST_LEN unsigned int
: The number of data transfers (beats) in this burst.

DATA unsigned int
: The data transferred.

INNER_ATTRIBUTES unsigned int
: The inner attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

INST bool
: Is an instruction fetch.

LOCK enum
: Normal / Locked / Exclusive.

MASTER_ID unsigned int
: The AXI master ID.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS bool
: Is a nonsecure access.

OUTER_ATTRIBUTES unsigned int
: The outer attributes: bit0=write allocate, bit1=read allocate, bit2=cachable, bit3=bufferable, bit4=shareable.

PADDR unsigned int
: Physical address of access.

PRIV bool
: Is privileged access.

RESPONSE enum
: Whether the transaction was successful, or an error occurred.

USER_FLAGS unsigned int
: Core specific additional signals.

PRELOAD_DATA

Data preload from PLD instruction. Fields:

VADDR unsigned int
: Virtual address of the data that should be preloaded.

PRELOAD_INST

Instruction preload from PLI instruction. Fields:

VADDR unsigned int
: Virtual address of the instruction that should be preloaded.

PRE_CORE_LOAD

Trace just before a core load. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PRE_CORE_STORE

Trace just before a core store. Fields:

CURRENT_TIME unsigned int
: The core's current time, as simulated time plus local time.

LOCAL_TIME unsigned int
: The core's local time, relative to the current quantum.

PSTATE_PACTIVE_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

PSTATE_PACTIVE_OPMODE_TRANSITION_DENIED

Trace the denied pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_POWER_TRANSITION

Trace pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

PSTATE_PACTIVE_POWER_TRANSITION_DENIED

Trace the denied pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Power mode after the transition

OLD_STATE enum
: Power mode before the transition

REASON string
: Reason for the transition being denied

PSTATE_PACTIVE_THREAD_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions for threads on PChannel. Fields:

DEVPACTIVE unsigned int
: Integer representation of DEVPACTIVE[] signals

NEW_STATE enum
: Operating mode after the transition

OLD_STATE enum
: Operating mode before the transition

RUN_STATE

Run state transition. Fields:

INST_COUNT unsigned int
: Ticks count at point of transition.

NEW enum
: New run state.

OLD enum
: Old run state.

SEMIHOSTING_CALL

Call of a semihost function occurred. Fields:

PC unsigned int
: The program counter after the semihosting call.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SEMIHOSTING_PRECALL

About to call semihost. Fields:

EL2 bool
: Use EL2 translation regime.

NS bool
: Is Non-Secure.

REG_WIDTH unsigned int
: The current register width in bytes.

SIGNAL

External signal state change. Fields:

SIGNAL enum
: Signal that changed

STATE bool
: Signal asserted state

START_COMPILE

Compilation started. Fields:

VADDR unsigned int
: Instruction where compilation begins.

STG_MULTIPLE_MICRO_OPS_ADDRESS_RANGE

Address range for MTE stores with multiple uops. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

STORE_MULTIPLE_REGS_ADDRESS_RANGE

Address range for stores with multiple registers. Fields:

ADDR_END unsigned int
: Final address.

ADDR_START unsigned int
: Starting address.

SUBOPTIMAL_LDST_RETIRED

suboptimal load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1: half-word, 2: word ...)

OPTIMAL_ALIGN_SELECTOR unsigned int
: Selector to decide the optimal alignment size (1: access width, 2: 4-bytes, 3: 8-bytes, ..., 12: 4-kbytes)

VADDR unsigned int
: The virtual address of the access

SYNC

Called for every synchronization, that is every quantum break. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INST_COUNT unsigned int
: The instruction count of this CPU.

LOCAL_QUANTUM unsigned int
: The local quantum of this CPU.

LOCAL_TIME unsigned int
: The local time of this CPU.

SYNC_REASON enum
: Reason for sync.

SYSCALL

System call instruction executed. Fields:

IMM unsigned int
: Immediate value of the system call instruction.

TYPE enum
: System call type.

VADDR unsigned int
: Instruction that caused the system call.

SYSREG_READ

Trace reads from system registers. Fields:

REG_NAME enum
: Register number

VALUE unsigned int
: The value read.

SYSREG_UPDATE32

Triggers when the system updates a register. Fields:

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

SYSREG_UPDATE64

Triggers when the system updates a register. Fields:

REG enum
: Register number.

UNKNOWN unsigned int
: Bits of the register which became unknown.

VALUE unsigned int
: Value written to the register.

SYSREG_WRITE

Trace write to system registers. Fields:

REG_ACCESS enum
: Register access status

REG_NAME enum
: Register number

UPDATED_VALUE unsigned int
: Updated value of the register now it has been written.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VALUE unsigned int
: Value written to the register.

UNALIGNED_LDST_RETIRED

Processor unaligned load/store. Fields:

ACCESS_SIZE unsigned int
: Log2 of the access width in bytes used for alignment checking (1=>half-word, 2->word ...)

NUMBER_OF_BEATS unsigned int
: Number of accesses (beats) in this burst (Total size in bytes is ACCESS_SIZE * NUMBER_OF_BEATS)

VADDR unsigned int
: The virtual address of the access

WAYPOINT

Signals end of basic block execution. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

ISET enum
: Origin instructions set

IS_COND bool
: Indicates if this is a conditional waypoint.

PC unsigned int
: Origin address (or 0 if unavailable)

TAKEN bool
: Indicates if this waypoint was taken.

TARGET unsigned int
: Destination address

TARGET_ISET enum
: Destination instructions set

WFE_END

WFE ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFE.

WFE_EVENT_REGISTER

WFE event register status: set/clear, reason. Fields:

INST_COUNT unsigned int
: Ticks count.

REASON enum
: Reason for set/clear. Only REASON==0 (Cleared by WFE) clears the bit, all other reasons set it.

WFE_IGNORED

WFE ignored. Fields:

EVENT bool
: This WFE was ignored because the event register was set.

INST_COUNT unsigned int
: Ticks count when ignoring WFE.

TRAPPED bool
: This WFE was trapped.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

WFE_START

WFE entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFE.

WFI_END

WFI ended. Fields:

INST_COUNT unsigned int
: Ticks count when leaving WFI.

WFI_IGNORED

WFI ignored. Fields:

DISABLED bool
: This WFI was ignored because WFI is disabled.

INST_COUNT unsigned int
: Ticks count when ignoring WFI.

REASON enum
: specifies reason why WFI trace was ignored

TRAPPED bool
: This WFI was trapped.

WFI_START

WFI entered. Fields:

INST_COUNT unsigned int
: Ticks count when entering WFI.

WFI_WAKEUP

WFI wakeup. Fields:

INST_COUNT unsigned int
: Ticks count when WFI wakeup occurred.

REASON enum
: Reason for wakeup.

XPSR

Changes to the xPSR register. Fields:

CORE_NUM unsigned int
: Core number in a multi processor.

OLD_VALUE unsigned int
: The old xPSR value

UNKNOWN unsigned int
: Bits within the register that have unknown value.

VALUE unsigned int
: The new xPSR value

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.48 ARMv8Cluster - trace

This section describes the trace sources.

CHECKPOINT_MESSAGE

Report error messages from the checkpointing process. Fields:

message `string`
: Message contents.

CHECKPOINT_RESTORE_END

Checkpoint restore completed.

CHECKPOINT_RESTORE_START

Checkpoint restore about to start.

CHECKPOINT_SAVE_END

Checkpoint save completed.

CHECKPOINT_SAVE_START

Checkpoint save about to start.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.49 AsyncCacheFlushUnit - trace

This section describes the trace sources.

STATE_CHANGE

Change in state of the AsyncCacheFlushUnit state machine. Fields:

STATE enum
: New state

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.50 Base_PowerController - trace

This section describes the trace sources.

PWRC_register_read

A register read occurred. Fields:

data unsigned int
: The data returned if not an error

error bool
: An error is being returned

ns bool
: The security state of the access

offset enum
: The offset of the register access

PWRC_register_write

A register write occurred. Fields:

data unsigned int
: The data written if not an error

error bool
: An error is being returned

ns bool
: The security state of the access

offset enum
: The offset of the register access

PWRC_signal_input

signal inputs. Fields:

index unsigned int
: index into port array

signal enum
: input signal

value bool
: whether signal is asserted or not

PWRC_signal_output

signal outputs. Fields:

index unsigned int
: index into port array

signal enum
: output signal

value bool
: whether signal is asserted or not

PWRC_sys_status_update

SYS_STATUS register is updated. Fields:

affinity unsigned int
: affinity level identifier of the core which is changing

data unsigned int
: new value of SYS_STATUS

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

index `unsigned int`
pin index to which this core is wired

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.51 CCI400 - trace

This section describes the trace sources.

CCI_REGISTER_READ

A valid read from the register file. (Access violations are not traced.). Fields:

Offset unsigned int
: Offset into the register block

Value unsigned int
: Value read

CCI_REGISTER_WRITE

A valid write to the register file. (Access violations are not traced). Fields:

Offset unsigned int
: Offset into the register block

Value unsigned int
: Value written

CHECKPOINT_MESSAGE

Report error messages from the checkpointing process. Fields:

message string
: Message contents.

CHECKPOINT_RESTORE_END

Checkpoint restore completed.

CHECKPOINT_RESTORE_START

Checkpoint restore about to start.

CHECKPOINT_SAVE_END

Checkpoint save completed.

CHECKPOINT_SAVE_START

Checkpoint save about to start.

read_for_0_came_from_snoop

A read from master 0 was satisfied by a snoop from an upstream port. Fields:

Bus address unsigned int
: The start bus address of the read transaction

Is non secure bool
: Is the transaction for the NS world?

read_for_1_came_from_snoop

A read from master 1 was satisfied by a snoop from an upstream port. Fields:

Bus address unsigned int
: The start bus address of the read transaction

Is non secure bool
: Is the transaction for the NS world?

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

read_for_2_came_from_snoop

A read from master 2 was satisfied by a snoop from an upstream port. Fields:

Bus address unsigned int
: The start bus address of the read transaction

Is non secure bool
: Is the transaction for the NS world?

read_for_3_came_from_snoop

A read from master 3 was satisfied by a snoop from an upstream port. Fields:

Bus address unsigned int
: The start bus address of the read transaction

Is non secure bool
: Is the transaction for the NS world?

read_for_4_came_from_snoop

A read from master 4 was satisfied by a snoop from an upstream port. Fields:

Bus address unsigned int
: The start bus address of the read transaction

Is non secure bool
: Is the transaction for the NS world?

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.52 CCI500 - trace

This section describes the trace sources.

ArchMsg.Error.reset_state_of_upstream_is_in_reset_but_trying_to_enable_snoops_and or_dvm

The upstream system is known to be in reset but we are trying to enable snoops/DVM to the port. In a real system, this could deadlock as snoop transactions could get routed to the port unless there were absolutely no shared transactions in flight or made. In any event, it is highly unlikely that the programmer intended this. Fields:

is_dvm_enable bool
: Is DVM enable (rather than snoop enable).

upstream_port_index unsigned int
: The upstream port index of the reset signal.

ArchMsg.Error.sw_reset

An error occurred in managing the reset of upstream systems whose entrance and exit of the coherency domain is managed by software. Fields:

why string
: Why the problem occurred.

ArchMsg.Warning.acchannelensx_enables_an_upstream_port_with_no_snoop_channel

ACCHANNELENSx should tell us which upstream ports have a connected snoop and DVM channels. The model has detected that an enabled port does not have a snoop/DVM channel. This will be silently ignored in the model, but might have deadlocked the hardware. Fields:

acchannelensx unsigned int
: The value of ACCHANNELENSx that is being used.

upstream_port_index unsigned int
: Upstream port index of offending port

ArchMsg.Warning.odd_programming_trying_to_enable_upstream_port

DESCRIPTION: The software tried to enable %{which} messages on port %{upstream_port_index} that is not capable of receiving these messages, for example by being disallowed by ACCHANNELENS% {upstream_port_index} or entirely disconnected. Fields:

acchannelensx unsigned int
: The ACCHANNELENSx bitmap of upstream ports.

bitmap_of_connected_upstream_ports unsigned int
: This is the bitmap of upstream ports that are actually connected.

bitmap_of_dvm_upstream_ports unsigned int
: The bitmap of DVM-capable upstream ports.

bitmap_of_full_ace_upstream_ports unsigned int
: The bitmap of snoop-capable upstream ports.

upstream_port_index unsigned int
: The upstream port index of the port

which enum
: Which type of enable we were trying to enable

ArchMsg.Warning.reset_state_of_upstream_port_driven_after_first_transaction_seen

All signals should be driven at simulation reset so that both sides of the connection agree on the value of the wire. This trace source means that the driver of the signal did not do this. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

upstream_port_index unsigned int
: The upstream port index of the reset signal.

value bool
: The value driven.

ArchMsg.Warning.reset_state_of_upstream_system_changed_whilst_a_snoop_change_is_pending

If snoop change request is pending, and the upstream system is reset during that process then the system could deadlock if there is any possibility of a shared transaction being in flight at the time. The upstream port might have an outstanding snoop request when it is reset and the interconnect would wait indefinitely for the response. Fields:

upstream_port_index unsigned int
: The upstream port index of the reset signal.

value bool
: The value driven.

why enum
: Why this message is emitted.

ArchMsg.Warning.shared_transaction_received_from_upstream_port_where_acchannelensx_is_disabled

ACCHANNELENSx tells the interconnect which upstream ports do not have connected snoop channels. This warns when a shared transaction is received from an upstream ACE port that doesn't have its snoop channel enabled by ACCHANNELENSx and there are other ports. This may be OK if the upstream system does not accept Snoop/DVM and so is purely an ACE-Lite system, or the upstream systems are truly separate shareability domains. However, this might also mean that you have completely forgotten to program the interconnect or use the parameter force_on_from_start. This will only be warned _once_ between resets of the interconnect. Fields:

upstream_port_index unsigned int
: Upstream port index of offending port

allowed_register_access

An allowed register access to the register file. Fields:

data unsigned int
: The data read/written. If this is a read, it is only valid if result is ok.

is_debug bool
: Is the transaction a debug transaction?

is_read bool
: Is the transaction a read transaction?

ns bool
: The transaction is non-secure

register enum
: The register of the transaction

change_dvm_disabled_ports

This reports a bitmap of which ports are disabled to receive DVM messages. Fields:

new_bitmap unsigned int
: The new bitmap of disabled ports

new_bitmap_list unsigned int
: A list of port numbers that were disabled.

old_bitmap unsigned int
: The old bitmap of disabled ports

old_bitmap_list unsigned int
: A list of port numbers that were disabled.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

change_upstream_disabled_ports

This reports a bitmap of which upstream ports of the cache are _disabled_ and so we won't sends snoop requests to them. Fields:

new_bitmap unsigned int
: The bitmap of the first 32 upstream ports that are disabled.

new_bitmap_list unsigned int
: A list of port numbers that were disabled.

old_bitmap unsigned int
: The bitmap of the first 32 upstream ports that are disabled.

old_bitmap_list unsigned int
: A list of port numbers that were disabled.

configuration_pins

One of the configuration pins changed. In the RTL these are sampled at reset, but in the model, we sample it at first transaction. If it occurs after this point then it is traced but the field 'ignored' will be true and the value reported will be what it wanted to change to rather than the value it actually is. Fields:

ignored bool
: The value was ignored because it occurred after seeing a transaction.

value unsigned int
: The value it wants to change to.

which enum
: Which configuration pin changed.

debug_enables

The signals/parameters that cause whether events are counted or not. Fields:

dbgen bool
: DBGEN value

niden bool
: NIDEN value

non_secure_traceable bool
: Non-secure events are traceable

secure_traceable bool
: Secure events are traceable

spiden bool
: SPIDEN value

spniden bool
: SPNIDEN value

errirq

This is the error interrupt signal that is generated when there are unacknowledged imprecise errors. Fields:

value bool
: The value it is driven to.

evntcntoverflow

The event counter overflow interrupt lines. Fields:

counter_id unsigned int
: The ID of the counter that this relates to.

value bool
: The value of the interrupt line

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

force_on_from_start

A commentary on what force_on_from_start is doing. Fields:

acchannelen_dvm unsigned int
: The bitmap of which upstream ports are capable of receiving DVM messages. If the appropriate bit is clear in this register then we won't enable it.

acchannelen_snoop unsigned int
: The bitmap of which upstream ports are capable of receiving snoop messages. If the appropriate bit is clear in this register then we won't enable it.

enable bool
: Whether we are trying to enable the upstream port or not.

reset_state_of_upstream_system enum
: What the interconnect knows about the reset state of the upstream system.

upstream_port_index unsigned int
: The upstream port index.

why enum
: Why the value changed

ignored_register_access

The register access is ignored for some reason. Fields:

address enum
: The address.

is_debug bool
: Is the access a debug access?

is_read bool
: Is the access a read?

ns bool
: The security state of the transaction.

why enum
: Why the access was ignored.

imprecise_error_received

An imprecise error was received for the specified transaction. Fields:

address unsigned int
: Address of transaction

downstream_port_index unsigned int
: The downstream port index that the transaction went out.

ns bool
: Is the transaction non-secure

pmu_overflow

The performance monitor counter overflowed. Fields:

counter_id unsigned int
: The ID of the counter that this relates to.

event_counting unsigned int
: The event that the counter is counting.

new_value bool
: The new value of the overflowed flag.

old_value bool
: The old value of the overflowed flag.

pmu_programmed

A programmed change to the performance monitor unit counter. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

counter_id unsigned int
: The ID of the counter that this relates to.

current_count unsigned int
: The current value of the counter

enabled bool
: Is the counter enabled?

event_counting unsigned int
: The event that the counter is counting.

why enum
: why

register_access

Read/Write of the register file. Fields:

address unsigned int
: The address of the transaction

data unsigned int
: The data read/written. If this is a read, it is only valid if result is ok.

is_debug bool
: Is the transaction a debug transaction?

is_read bool
: Is the transaction a read transaction?

ns bool
: Is the transaction non-secure?

number_of_beats unsigned int
: The number of beats in the transaction

result enum
: The result of the transaction

width_in_bytes unsigned int
: Width of the transaction in bytes.

reset_state_of_upstream_port

The pins reset_state_of_upstream_port[N] are intended to be use as checks by the interconnect that the programmer has disabled snoops and DVMs to upstream ports whose systems are in reset. If you connect these ports to a signal that indicates if the upstream port is in reset then the interconnect can perform this check. This trace source is the trace of those reset state changes. Fields:

upstream_port_index unsigned int
: The upstream port index of the reset signal.

value bool
: The value driven.

upstream_port_with_an_unsynced_snoop_change_pending

When a change to whether to send snoop requests or DVM messages to a port is setup then it takes a finite period of time to take effect and the SW should poll for completion. This tells you when an upstream port has a snoop change pending on it that hasn't been polled for completion. Fields:

comment enum
: Comment

pending bool
: Whether the change is pending or not

upstream_port_index unsigned int
: Upstream port index

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.53 CCI550 - trace

This section describes the trace sources.

ArchMsg.Error.reset_state_of_upstream_is_in_reset_but_trying_to_enable_snoops_and or_dvm

The upstream system is known to be in reset but we are trying to enable snoops/DVM to the port. In a real system, this could deadlock as snoop transactions could get routed to the port unless there were absolutely no shared transactions in flight or made. In any event, it is highly unlikely that the programmer intended this. Fields:

is_dvm_enable bool
: Is DVM enable (rather than snoop enable).

upstream_port_index unsigned int
: The upstream port index of the reset signal.

ArchMsg.Error.sw_reset

An error occurred in managing the reset of upstream systems whose entrance and exit of the coherency domain is managed by software. Fields:

why string
: Why the problem occurred.

ArchMsg.Warning.acchannelensx_enables_an_upstream_port_with_no_snoop_channel

ACCHANNELENSx should tell us which upstream ports have a connected snoop and DVM channels. The model has detected that an enabled port does not have a snoop/DVM channel. This will be silently ignored in the model, but might have deadlocked the hardware. Fields:

acchannelensx unsigned int
: The value of ACCHANNELENSx that is being used.

upstream_port_index unsigned int
: Upstream port index of offending port

ArchMsg.Warning.odd_programming_trying_to_enable_upstream_port

DESCRIPTION: The software tried to enable %{which} messages on port %{upstream_port_index} that is not capable of receiving these messages, for example by being disallowed by ACCHANNELENS% {upstream_port_index} or entirely disconnected. Fields:

acchannelensx unsigned int
: The ACCHANNELENSx bitmap of upstream ports.

bitmap_of_connected_upstream_ports unsigned int
: This is the bitmap of upstream ports that are actually connected.

bitmap_of_dvm_upstream_ports unsigned int
: The bitmap of DVM-capable upstream ports.

bitmap_of_full_ace_upstream_ports unsigned int
: The bitmap of snoop-capable upstream ports.

upstream_port_index unsigned int
: The upstream port index of the port

which enum
: Which type of enable we were trying to enable

ArchMsg.Warning.reset_state_of_upstream_port_driven_after_first_transaction_seen

All signals should be driven at simulation reset so that both sides of the connection agree on the value of the wire. This trace source means that the driver of the signal did not do this. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

upstream_port_index unsigned int
: The upstream port index of the reset signal.

value bool
: The value driven.

ArchMsg.Warning.reset_state_of_upstream_system_changed_whilst_a_snoop_change_is_pending

If snoop change request is pending, and the upstream system is reset during that process then the system could deadlock if there is any possibility of a shared transaction being in flight at the time. The upstream port might have an outstanding snoop request when it is reset and the interconnect would wait indefinitely for the response. Fields:

upstream_port_index unsigned int
: The upstream port index of the reset signal.

value bool
: The value driven.

why enum
: Why this message is emitted.

ArchMsg.Warning.shared_transaction_received_from_upstream_port_where_acchannelensx_is_disabled

ACCHANNELENSx tells the interconnect which upstream ports do not have connected snoop channels. This warns when a shared transaction is received from an upstream ACE port that doesn't have its snoop channel enabled by ACCHANNELENSx and there are other ports. This may be OK if the upstream system does not accept Snoop/DVM and so is purely an ACE-Lite system, or the upstream systems are truly separate shareability domains. However, this might also mean that you have completely forgotten to program the interconnect or use the parameter force_on_from_start. This will only be warned _once_ between resets of the interconnect. Fields:

upstream_port_index unsigned int
: Upstream port index of offending port

allowed_register_access

An allowed register access to the register file. Fields:

data unsigned int
: The data read/written. If this is a read, it is only valid if result is ok.

is_debug bool
: Is the transaction a debug transaction?

is_read bool
: Is the transaction a read transaction?

ns bool
: The transaction is non-secure

register enum
: The register of the transaction

change_dvm_disabled_ports

This reports a bitmap of which ports are disabled to receive DVM messages. Fields:

new_bitmap unsigned int
: The new bitmap of disabled ports

new_bitmap_list unsigned int
: A list of port numbers that were disabled.

old_bitmap unsigned int
: The old bitmap of disabled ports

old_bitmap_list unsigned int
: A list of port numbers that were disabled.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

change_upstream_disabled_ports

This reports a bitmap of which upstream ports of the cache are _disabled_ and so we won't sends snoop requests to them. Fields:

new_bitmap unsigned int
: The bitmap of the first 32 upstream ports that are disabled.

new_bitmap_list unsigned int
: A list of port numbers that were disabled.

old_bitmap unsigned int
: The bitmap of the first 32 upstream ports that are disabled.

old_bitmap_list unsigned int
: A list of port numbers that were disabled.

configuration_pins

One of the configuration pins changed. In the RTL these are sampled at reset, but in the model, we sample it at first transaction. If it occurs after this point then it is traced but the field 'ignored' will be true and the value reported will be what it wanted to change to rather than the value it actually is. Fields:

ignored bool
: The value was ignored because it occurred after seeing a transaction.

value unsigned int
: The value it wants to change to.

which enum
: Which configuration pin changed.

debug_enables

The signals/parameters that cause whether events are counted or not. Fields:

dbgen bool
: DBGEN value

niden bool
: NIDEN value

non_secure_traceable bool
: Non-secure events are traceable

secure_traceable bool
: Secure events are traceable

spiden bool
: SPIDEN value

spniden bool
: SPNIDEN value

errirq

This is the error interrupt signal that is generated when there are unacknowledged imprecise errors. Fields:

value bool
: The value it is driven to.

evntcntoverflow

The event counter overflow interrupt lines. Fields:

counter_id unsigned int
: The ID of the counter that this relates to.

value bool
: The value of the interrupt line

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

force_on_from_start

A commentary on what force_on_from_start is doing. Fields:

acchannelen_dvm unsigned int
: The bitmap of which upstream ports are capable of receiving DVM messages. If the appropriate bit is clear in this register then we won't enable it.

acchannelen_snoop unsigned int
: The bitmap of which upstream ports are capable of receiving snoop messages. If the appropriate bit is clear in this register then we won't enable it.

enable bool
: Whether we are trying to enable the upstream port or not.

reset_state_of_upstream_system enum
: What the interconnect knows about the reset state of the upstream system.

upstream_port_index unsigned int
: The upstream port index.

why enum
: Why the value changed

ignored_register_access

The register access is ignored for some reason. Fields:

address enum
: The address.

is_debug bool
: Is the access a debug access?

is_read bool
: Is the access a read?

ns bool
: The security state of the transaction.

why enum
: Why the access was ignored.

imprecise_error_received

An imprecise error was received for the specified transaction. Fields:

address unsigned int
: Address of transaction

downstream_port_index unsigned int
: The downstream port index that the transaction went out.

ns bool
: Is the transaction non-secure

pmu_overflow

The performance monitor counter overflowed. Fields:

counter_id unsigned int
: The ID of the counter that this relates to.

event_counting unsigned int
: The event that the counter is counting.

new_value bool
: The new value of the overflowed flag.

old_value bool
: The old value of the overflowed flag.

pmu_programmed

A programmed change to the performance monitor unit counter. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

counter_id unsigned int
: The ID of the counter that this relates to.

current_count unsigned int
: The current value of the counter

enabled bool
: Is the counter enabled?

event_counting unsigned int
: The event that the counter is counting.

why enum
: why

register_access

Read/Write of the register file. Fields:

address unsigned int
: The address of the transaction

data unsigned int
: The data read/written. If this is a read, it is only valid if result is ok.

is_debug bool
: Is the transaction a debug transaction?

is_read bool
: Is the transaction a read transaction?

ns bool
: Is the transaction non-secure?

number_of_beats unsigned int
: The number of beats in the transaction

result enum
: The result of the transaction

width_in_bytes unsigned int
: Width of the transaction in bytes.

reset_state_of_upstream_port

The pins reset_state_of_upstream_port[N] are intended to be use as checks by the interconnect that the programmer has disabled snoops and DVMs to upstream ports whose systems are in reset. If you connect these ports to a signal that indicates if the upstream port is in reset then the interconnect can perform this check. This trace source is the trace of those reset state changes. Fields:

upstream_port_index unsigned int
: The upstream port index of the reset signal.

value bool
: The value driven.

upstream_port_with_an_unsynced_snoop_change_pending

When a change to whether to send snoop requests or DVM messages to a port is setup then it takes a finite period of time to take effect and the SW should poll for completion. This tells you when an upstream port has a snoop change pending on it that hasn't been polled for completion. Fields:

comment enum
: Comment

pending bool
: Whether the change is pending or not

upstream_port_index unsigned int
: Upstream port index

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.54 CCNCache - trace

This section describes the trace sources.

ALLOC_LINEFILL

The system allocated a complete line fill into the RAMs. This happens when the cache has to read the data for a line fill from potentially upstream or downstream. Fields:

ENTRY_INDEX `unsigned int`
: The entry index the line has been loaded into.

MASTER_ID `unsigned int`
: The master ID for the associated transaction.

MEMORY_ATTRIBUTES `unsigned int`
: The memory attributes used for the allocation. bits[3:0] are the inner ACACHE, bits[7:4] are the outer ACACHE, bits[9:8] are ADOMAIN, bits[12:10] are APROT, the rest are undocumented.

NS_ADDR `unsigned int`
: The security world and address of the first byte of the line, the MSB encodes the NS bit.

ALLOC_WRITE

The system allocated a complete line write into the RAMs. This happens when the cache receives a complete cache line write. It need not read from anywhere. Fields:

ENTRY_INDEX `unsigned int`
: The entry index the line has been written into.

MEMORY_ATTRIBUTES `unsigned int`
: The memory attributes used for the allocation. bits[3:0] are the inner ACACHE, bits[7:4] are the outer ACACHE, bits[9:8] are ADOMAIN, bits[12:10] are APROT, the rest are undocumented.

NS_ADDR `unsigned int`
: The security world and address of the first byte of the line, the MSB encodes the NS bit.

ATOMIC_SLAVE_ACCESS

Information on the atomic operation at slave side. Fields:

ACCESS_RESULT `enum`
: Result of atomic access

COMPARE_VALUE `unsigned int`
: Value to be compared with memory by CAS

LOAD_VALUE `unsigned int`
: Value to be loaded from memory as a result of atomic operation

MASTER `unsigned int`
: ID of master which initiated the transaction

NS `bool`
: Transaction is Non-Secure

OPERAND_VALUE `unsigned int`
: Incoming operand value provided by core

OPERATION `enum`
: Type of atomic operation

PADDR `unsigned int`
: Physical address

STORE_VALUE `unsigned int`
: Value to be stored to memory as a result of atomic operation

ArchMsg.Error.cached_attributes_mismatch

Attributes differ between page and line in cache from same page. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LINEATTR unsigned int
: Attributes of the cache allocated lines

LINES unsigned int
: Number of cache lines already allocated

NS_ADDR unsigned int
: The security world and address of the first byte of the page, the MSB encodes the NS bit.

TXATTR unsigned int
: Attributes of the transaction

ArchMsg.Error.cached_attributes_mismatch#lineinfo

Information about allocated lines in page. Fields:

PADDR unsigned int
: address of this cache line

TAG unsigned int
: tag data of this cache line

ArchMsg.Warning.ns_s_dirty_hit

Cache hit at S and NS versions of the same physical address. Fields:

INDEX0 unsigned int
: The cache line index of the affected cache line

INDEX1 unsigned int
: The cache line index of the other regime cache line

PADDR unsigned int
: The address of the affected cache line

TAG0 unsigned int
: The tag data of the affected cache line

TAG1 unsigned int
: The tag data of the other regime cache line

CACHE_ACCESS

Number of cache lookups.

CACHE_MISS

cache access cache miss.

CACHE_READ_HIT

Read access cache hit. Note that this trace source causes a large slowdown in the simulation. Fields:

IS_PRELOAD enum
: Is the access a preload

IS_SHARED enum
: Is the access shared

LATENCY unsigned int
: Cache latency in ticks per access

MASTER_ID unsigned int
: Indicates the master associated with this monitor

CACHE_READ_MISS

Read access cache miss and cache miss latency. Fields:

IS_PRELOAD enum
: Is the access a preload

IS_SHARED enum
: Is the access shared

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LATENCY unsigned int
: Cache miss latency in ticks per access

MASTER_ID unsigned int
: Indicates the master associated with this monitor

CACHE_SNOOP BROADCAST

cache snoop accesses.

CACHE_WRITE_HIT

Write access cache hit. Note that this trace source causes a large slowdown in the simulation. Fields:

IS_PRELOAD enum
: Is the access a preload

IS_SHARED enum
: Is the access shared

LATENCY unsigned int
: Cache latency in ticks per access

MASTER_ID unsigned int
: Indicates the master associated with this monitor

CACHE_WRITE_MISS

Write access cache miss and cache miss latency. Fields:

IS_PRELOAD enum
: Is the access a preload

IS_SHARED enum
: Is the access shared

LATENCY unsigned int
: Cache miss latency in ticks per access

MASTER_ID unsigned int
: Indicates the master associated with this monitor

ENTRY_BECOMES_INVALID

An entry is now invalid. The cause could be any reason. Fields:

ENTRY_INDEX unsigned int
: The entry index that has become invalid.

ENTRY_SET_DIRTY

Entry change, clean/dirty. Fields:

ENTRY_INDEX unsigned int
: The entry index that has become clean or dirty.

STATUS unsigned int
: True if the line was made dirty

ERROR_MIXED_ATTRIBUTES_LINE

Mismatched line and transaction attributes. Fields:

ENTRY_INDEX unsigned int
: The entry index that has collided.

NS_ADDR unsigned int
: The security world and address of the first byte of the line, the MSB encodes the NS bit.

ERROR_MIXED_ATTRIBUTES_PAGE

Attributes differ between page and lines in cache from the same page. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS_ADDR unsigned int
: The security world and address of the first byte of the page, the MSB encodes the NS bit.

change_dvm_disabled_ports

This reports a bitmap of which ports are disabled to receive DVM messages. Fields:

new_bitmap unsigned int
: The new bitmap of disabled ports

new_bitmap_list unsigned int
: A list of port numbers that were disabled.

old_bitmap unsigned int
: The old bitmap of disabled ports

old_bitmap_list unsigned int
: A list of port numbers that were disabled.

change_upstream_disabled_ports

This reports a bitmap of which upstream ports of the cache are _disabled_ and so we won't sends snoop requests to them. Fields:

new_bitmap unsigned int
: The bitmap of the first 32 upstream ports that are disabled.

new_bitmap_list unsigned int
: A list of port numbers that were disabled.

old_bitmap unsigned int
: The bitmap of the first 32 upstream ports that are disabled.

old_bitmap_list unsigned int
: A list of port numbers that were disabled.

entry_after_far_atomic

A cache line has had a far atomic applied to it. Fields:

atomic_operation enum
: The operation that operated on this line.

be bool
: The far atomic's big-endian flag.

data unsigned int
: The data in the cache line.

entry_begin_address unsigned int
: The start address of the entry.

entry_index unsigned int
: The entry index in the cache.

incoming_data unsigned int
: The incoming data from the far atomic. For CAS operations then this is compare-value and then the replacement-value

master_id unsigned int
: The master ID for the associated transaction

ns enum
: The security state of the line

prior_data unsigned int
: The data that will be returned as the 'prior' data of the far atomic.

trans_begin_address unsigned int
: The start address of the transaction.

entry_after_read

A cache line has been read. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ace_operation enum
: The ACE operation that operated on this line.

data unsigned int
: The data in the cache line.

dirty bool
: Is the line ACE-dirty.

entry_begin_address unsigned int
: The start address of the entry.

entry_index unsigned int
: The entry index in the cache.

ns enum
: The security state of the line

snoop bool
: This transaction is a snoop transaction

trans_begin_address unsigned int
: The start address of the transaction.

unique bool
: Is the line ACE-unique.

entry_after_write

A cache line has been written. Fields:

ace_operation enum
: The ACE operation that operated on this line.

data unsigned int
: The data in the cache line.

entry_begin_address unsigned int
: The start address of the entry.

entry_index unsigned int
: The entry index in the cache.

ns enum
: The security state of the line

trans_begin_address unsigned int
: The start address of the transaction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.55 CCNRegisterSet - trace

This section describes the trace sources.

ArchMsg.Warning.CCNRegister.MemoryMapped_ReadReserved

Fields:

OFFSET unsigned int
: Offset of address within the CCN module.

ArchMsg.Warning.CCNRegister.MemoryMapped_ReadWriteOnlyReg

Fields:

OFFSET unsigned int
: Offset of address within the CCN module.

REG_NAME string
: Name of the register being read.

ArchMsg.Warning.CCNRegister.MemoryMapped_WriteReadOnlyReg

Fields:

OFFSET unsigned int
: Offset of address within the CCN module.

REG_NAME string
: Name of the register being written.

VALUE unsigned int
: Value written.

ArchMsg.Warning.CCNRegister.MemoryMapped_WriteReserved

Fields:

OFFSET unsigned int
: Offset of address within the CCN module.

VALUE unsigned int
: Value written.

CCNRegister.MemoryMapped_Read

Trace read from a CCN memory-mapped register. Fields:

NS bool
: Access is non-secure.

OFFSET unsigned int
: Offset of address within the CCN module.

REG_NAME string
: Name of the register being read.

VALUE unsigned int
: Value returned.

CCNRegister.MemoryMapped_Write

Trace write to a CCN memory-mapped register. Fields:

NS bool
: Access is non-secure.

OFFSET unsigned int
: Offset of address within the CCN module.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

REG_NAME `string`
: Name of the register being written.

UPDATED_VALUE `unsigned int`
: New value read back from the register; for write-only registers this will be equal to VALUE.

VALUE `unsigned int`
: Value written.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.56 CMN600 - trace

This section describes the trace sources.

ArchMsg.Warning.MemoryMapped_ReadReserved

Fields:

OFFSET unsigned int
: Offset of address within the CMN600.

ArchMsg.Warning.MemoryMapped_ReadWriteOnlyReg

Fields:

OFFSET unsigned int
: Offset of address within the CMN600.

REG_NAME string
: Name of the register being read.

ArchMsg.Warning.MemoryMapped_WriteReadOnlyReg

Fields:

OFFSET unsigned int
: Offset of address within the CMN600.

REG_NAME string
: Name of the register being written.

VALUE unsigned int
: Value written.

ArchMsg.Warning.MemoryMapped_WriteReserved

Fields:

OFFSET unsigned int
: Offset of address within the CMN600.

VALUE unsigned int
: Value written.

CMN600_Cache_Routing

Fields:

ADDR unsigned int
: Address of memory location being accessed.

CMN600_Downstream_Routing

Fields:

ADDR unsigned int
: Address of memory location being accessed.

PORT unsigned int
: Downstream port number to which access is routed.

TYPE string
: Downstream port type to which access is routed.

CMN600_Invalid_OCM_Region

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CACHE_REGION_END unsigned int
: End Address of memory location that belongs to the Cache.

CACHE_REGION_START unsigned int
: Start Address of memory location that belongs to the Cache.

OCM_REGION_END unsigned int
: End Address of OCM region.

OCM_REGION_START unsigned int
: Start Address of OCM region.

CMN600_OCM_Routing

Fields:

ADDR unsigned int
: Address of memory location being accessed.

OFFSET unsigned int
: OFFSET in OCM memory location to which access is routed.

HNFSAM_Memory_Map

Fields:

HNF_INDEX unsigned int
: Index of HNF.

MMAP string
: memory map of cmn600 hnfsam.

MemoryMapped_Read

Trace read from a memory-mapped register. Fields:

NS bool
: Access is non-secure.

OFFSET unsigned int
: Offset of address within the CMN600.

REG_NAME string
: Name of the register being read.

VALUE unsigned int
: Value returned.

MemoryMapped_Write

Trace write to a memory-mapped register. Fields:

NS bool
: Access is non-secure.

OFFSET unsigned int
: Offset of address within the CMN600.

REG_NAME string
: Name of the register being written.

UPDATED_VALUE unsigned int
: New value read back from the register; for write-only registers this will be equal to VALUE.

VALUE unsigned int
: Value written.

OCM_Regions

Fields:

REGIONS string
: memory regions allocated to OCM

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

RNSAM_Memory_Map

Fields:

MMAP string
: memory map of cmn600 rnsam.

RNSAM_Target_HNF

Fields:

ADDRESS unsigned int
: Transaction Address.

HNF_NUM unsigned int
: HNF Node Index to which the transaction is routed.

Warning.HNF_part_of_multiple_system_cache_group

Fields:

NODE_INDEX unsigned int
: HNF Node index which is part of multiple system cache group.

Warning.Hashing_enabled_for_cache_group

Fields:

HNF_INDEX unsigned int
: HNF Node index which is part of system cache group.

Warning.RNSAM_Not_Unique

Fields:

INDEX unsigned int
: Index of RN component. Index starts with RNFs and then the RNIs.

Warning.RNSAM_target_id already used

Fields:

NODEID unsigned int
: Node Id specified in target register.

NODE_TYPE enum
: Node type of node id specified in non_hash_tgt register.

Warning.RNSAM_target_id not matching target type

Fields:

NODEID unsigned int
: Node Id specified in non_hash_tgt register.

NODE_TYPE enum
: Node type of node id specified in non_hash_tgt register.

TARGET_TYPE enum
: Target Type specified in non_hash_region register.

Warning.event_smp_mode

Event not propagated SMP not enabled.

Warning.rnsam_and_cxrasam_mismatch

RNSAM and CXRASAM address regions don't match. You may need to review its configuration.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Warning.scg_hnf_count not valid

Fields:

SCG_HNF_COUNT unsigned int
: Number of HNF configured for the group.

SCG_REGION_NUM unsigned int
: System Cache Group region number.

event_propagation_downstream

Fields:

STATE unsigned int
: State being transmitted.

event_propagation_downstream_routing

Fields:

LINK unsigned int
: Target link.

PORT unsigned int
: Activated port.

STATE unsigned int
: State being transmitted.

event_propagation_upstream

Fields:

STATE unsigned int
: State being transmitted.

trace_add_cxra_memory_region

Fields:

END unsigned int
: End address of the memory region.

HAID unsigned int
: Memory region target HAID

START unsigned int
: Start address of the memory region.

trace_add_memory_region

Fields:

END unsigned int
: End address of the memory region.

START unsigned int
: Start address of the memory region.

TYPE enum
: Node type.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.57 CMN600CML - trace

This section describes the trace sources.

event_propagation_hub_routing

Fields:

CLIENT unsigned int
: Event initiator.

STATE enum
: State being transmitted.

trace_add_port_address_map

Fields:

CLIENT unsigned int
: Client port

END unsigned int
: End address of the memory region.

START unsigned int
: Start address of the memory region.

TARGET unsigned int
: Target port

trace_address_decode

Fields:

ADDRESS unsigned int
: Target Address

CLIENT unsigned int
: Client port

TARGET unsigned int
: Target port

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.58 CMN600Cache - trace

This section describes the trace sources.

ALLOC_LINEFILL

The system allocated a complete line fill into the RAMs. This happens when the cache has to read the data for a line fill from potentially upstream or downstream. Fields:

ENTRY_INDEX `unsigned int`
: The entry index the line has been loaded into.

MASTER_ID `unsigned int`
: The master ID for the associated transaction.

MEMORY_ATTRIBUTES `unsigned int`
: The memory attributes used for the allocation. bits[3:0] are the inner ACACHE, bits[7:4] are the outer ACACHE, bits[9:8] are ADOMAIN, bits[12:10] are APROT, the rest are undocumented.

NS_ADDR `unsigned int`
: The security world and address of the first byte of the line, the MSB encodes the NS bit.

ALLOC_WRITE

The system allocated a complete line write into the RAMs. This happens when the cache receives a complete cache line write. It need not read from anywhere. Fields:

ENTRY_INDEX `unsigned int`
: The entry index the line has been written into.

MEMORY_ATTRIBUTES `unsigned int`
: The memory attributes used for the allocation. bits[3:0] are the inner ACACHE, bits[7:4] are the outer ACACHE, bits[9:8] are ADOMAIN, bits[12:10] are APROT, the rest are undocumented.

NS_ADDR `unsigned int`
: The security world and address of the first byte of the line, the MSB encodes the NS bit.

ATOMIC_SLAVE_ACCESS

Information on the atomic operation at slave side. Fields:

ACCESS_RESULT `enum`
: Result of atomic access

COMPARE_VALUE `unsigned int`
: Value to be compared with memory by CAS

LOAD_VALUE `unsigned int`
: Value to be loaded from memory as a result of atomic operation

MASTER `unsigned int`
: ID of master which initiated the transaction

NS `bool`
: Transaction is Non-Secure

OPERAND_VALUE `unsigned int`
: Incoming operand value provided by core

OPERATION `enum`
: Type of atomic operation

PADDR `unsigned int`
: Physical address

STORE_VALUE `unsigned int`
: Value to be stored to memory as a result of atomic operation

ArchMsg.Error.cached_attributes_mismatch

Attributes differ between page and line in cache from same page. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LINEATTR unsigned int
: Attributes of the cache allocated lines

LINES unsigned int
: Number of cache lines already allocated

NS_ADDR unsigned int
: The security world and address of the first byte of the page, the MSB encodes the NS bit.

TXATTR unsigned int
: Attributes of the transaction

ArchMsg.Error.cached_attributes_mismatch#lineinfo

Information about allocated lines in page. Fields:

PADDR unsigned int
: address of this cache line

TAG unsigned int
: tag data of this cache line

ArchMsg.Warning.ns_s_dirty_hit

Cache hit at S and NS versions of the same physical address. Fields:

INDEX0 unsigned int
: The cache line index of the affected cache line

INDEX1 unsigned int
: The cache line index of the other regime cache line

PADDR unsigned int
: The address of the affected cache line

TAG0 unsigned int
: The tag data of the affected cache line

TAG1 unsigned int
: The tag data of the other regime cache line

CACHE_READ_HIT

Read access cache hit. Note that this trace source causes a large slowdown in the simulation. Fields:

IS_PRELOAD enum
: Is the access a preload

IS_SHARED enum
: Is the access shared

LATENCY unsigned int
: Cache latency in ticks per access

MASTER_ID unsigned int
: Indicates the master associated with this monitor

CACHE_READ_MISS

Read access cache miss and cache miss latency. Fields:

IS_PRELOAD enum
: Is the access a preload

IS_SHARED enum
: Is the access shared

LATENCY unsigned int
: Cache miss latency in ticks per access

MASTER_ID unsigned int
: Indicates the master associated with this monitor

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CACHE_WRITE_HIT

Write access cache hit. Note that this trace source causes a large slowdown in the simulation. Fields:

IS_PRELOAD enum
: Is the access a preload

IS_SHARED enum
: Is the access shared

LATENCY unsigned int
: Cache latency in ticks per access

MASTER_ID unsigned int
: Indicates the master associated with this monitor

CACHE_WRITE_MISS

Write access cache miss and cache miss latency. Fields:

IS_PRELOAD enum
: Is the access a preload

IS_SHARED enum
: Is the access shared

LATENCY unsigned int
: Cache miss latency in ticks per access

MASTER_ID unsigned int
: Indicates the master associated with this monitor

CMN600_REMAP_ABORT_TRACE

remap() returns abort port. Fields:

ABORT_REASON string
: Reason for routing to ABORT port in CMN600Interconnect cache

ENTRY_BECOMES_INVALID

An entry is now invalid. The cause could be any reason. Fields:

ENTRY_INDEX unsigned int
: The entry index that has become invalid.

ENTRY_SET_DIRTY

Entry change, clean/dirty. Fields:

ENTRY_INDEX unsigned int
: The entry index that has become clean or dirty.

STATUS unsigned int
: True if the line was made dirty

ERROR_MIXED_ATTRIBUTES_LINE

Mismatched line and transaction attributes. Fields:

ENTRY_INDEX unsigned int
: The entry index that has collided.

NS_ADDR unsigned int
: The security world and address of the first byte of the line, the MSB encodes the NS bit.

ERROR_MIXED_ATTRIBUTES_PAGE

Attributes differ between page and lines in cache from the same page. Fields:

NS_ADDR unsigned int
: The security world and address of the first byte of the page, the MSB encodes the NS bit.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

change_dvm_disabled_ports

This reports a bitmap of which ports are disabled to receive DVM messages. Fields:

new_bitmap unsigned int
: The new bitmap of disabled ports

new_bitmap_list unsigned int
: A list of port numbers that were disabled.

old_bitmap unsigned int
: The old bitmap of disabled ports

old_bitmap_list unsigned int
: A list of port numbers that were disabled.

change_upstream_disabled_ports

This reports a bitmap of which upstream ports of the cache are _disabled_ and so we won't sends snoop requests to them. Fields:

new_bitmap unsigned int
: The bitmap of the first 32 upstream ports that are disabled.

new_bitmap_list unsigned int
: A list of port numbers that were disabled.

old_bitmap unsigned int
: The bitmap of the first 32 upstream ports that are disabled.

old_bitmap_list unsigned int
: A list of port numbers that were disabled.

entry_after_far_atomic

A cache line has had a far atomic applied to it. Fields:

atomic_operation enum
: The operation that operated on this line.

be bool
: The far atomic's big-endian flag.

data unsigned int
: The data in the cache line.

entry_begin_address unsigned int
: The start address of the entry.

entry_index unsigned int
: The entry index in the cache.

incoming_data unsigned int
: The incoming data from the far atomic. For CAS operations then this is compare-value and then the replacement-value

master_id unsigned int
: The master ID for the associated transaction

ns enum
: The security state of the line

prior_data unsigned int
: The data that will be returned as the 'prior' data of the far atomic.

trans_begin_address unsigned int
: The start address of the transaction.

entry_after_read

A cache line has been read. Fields:

ace_operation enum
: The ACE operation that operated on this line.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

data unsigned int
: The data in the cache line.

dirty bool
: Is the line ACE-dirty.

entry_begin_address unsigned int
: The start address of the entry.

entry_index unsigned int
: The entry index in the cache.

ns enum
: The security state of the line

snoop bool
: This transaction is a snoop transaction

trans_begin_address unsigned int
: The start address of the transaction.

unique bool
: Is the line ACE-unique.

entry_after_write

A cache line has been written. Fields:

ace_operation enum
: The ACE operation that operated on this line.

data unsigned int
: The data in the cache line.

entry_begin_address unsigned int
: The start address of the entry.

entry_index unsigned int
: The entry index in the cache.

ns enum
: The security state of the line

trans_begin_address unsigned int
: The start address of the transaction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.59 CMN650 - trace

This section describes the trace sources.

ArchMsg.Warning.MemoryMapped_ReadReserved

Fields:

OFFSET unsigned int
: Offset of address within the CMN600.

ArchMsg.Warning.MemoryMapped_ReadWriteOnlyReg

Fields:

OFFSET unsigned int
: Offset of address within the CMN600.

REG_NAME string
: Name of the register being read.

ArchMsg.Warning.MemoryMapped_WriteReadOnlyReg

Fields:

OFFSET unsigned int
: Offset of address within the CMN600.

REG_NAME string
: Name of the register being written.

VALUE unsigned int
: Value written.

ArchMsg.Warning.MemoryMapped_WriteReserved

Fields:

OFFSET unsigned int
: Offset of address within the CMN600.

VALUE unsigned int
: Value written.

CMN600_Cache_Routing

Fields:

ADDR unsigned int
: Address of memory location being accessed.

CMN600_Downstream_Routing

Fields:

ADDR unsigned int
: Address of memory location being accessed.

PORT unsigned int
: Downstream port number to which access is routed.

TYPE string
: Downstream port type to which access is routed.

CMN600_Invalid_OCM_Region

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CACHE_REGION_END unsigned int
: End Address of memory location that belongs to the Cache.

CACHE_REGION_START unsigned int
: Start Address of memory location that belongs to the Cache.

OCM_REGION_END unsigned int
: End Address of OCM region.

OCM_REGION_START unsigned int
: Start Address of OCM region.

CMN600_OCM_Routing

Fields:

ADDR unsigned int
: Address of memory location being accessed.

OFFSET unsigned int
: OFFSET in OCM memory location to which access is routed.

HNFSAM_Memory_Map

Fields:

HNF_INDEX unsigned int
: Index of HNF.

MMAP string
: memory map of cmn600 hnfsam.

MemoryMapped_Read

Trace read from a memory-mapped register. Fields:

NS bool
: Access is non-secure.

OFFSET unsigned int
: Offset of address within the CMN600.

REG_NAME string
: Name of the register being read.

VALUE unsigned int
: Value returned.

MemoryMapped_Write

Trace write to a memory-mapped register. Fields:

NS bool
: Access is non-secure.

OFFSET unsigned int
: Offset of address within the CMN600.

REG_NAME string
: Name of the register being written.

UPDATED_VALUE unsigned int
: New value read back from the register; for write-only registers this will be equal to VALUE.

VALUE unsigned int
: Value written.

OCM_Regions

Fields:

REGIONS string
: memory regions allocated to OCM

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

RNSAM_Memory_Map

Fields:

MMAP string
: memory map of cmn600 rnsam.

RNSAM_Target_HNF

Fields:

ADDRESS unsigned int
: Transaction Address.

HNF_NUM unsigned int
: HNF Node Index to which the transaction is routed.

Warning.HNF_part_of_multiple_system_cache_group

Fields:

NODE_INDEX unsigned int
: HNF Node index which is part of multiple system cache group.

Warning.Hashing_enabled_for_cache_group

Fields:

HNF_INDEX unsigned int
: HNF Node index which is part of system cache group.

Warning.RNSAM_Not_Unique

Fields:

INDEX unsigned int
: Index of RN component. Index starts with RNFs and then the RNIs.

Warning.RNSAM_target_id already used

Fields:

NODEID unsigned int
: Node Id specified in target register.

NODE_TYPE enum
: Node type of node id specified in non_hash_tgt register.

Warning.RNSAM_target_id not matching target type

Fields:

NODEID unsigned int
: Node Id specified in non_hash_tgt register.

NODE_TYPE enum
: Node type of node id specified in non_hash_tgt register.

TARGET_TYPE enum
: Target Type specified in non_hash_region register.

Warning.event_smp_mode

Event not propagated SMP not enabled.

Warning.rnsam_and_cxrasam_mismatch

RNSAM and CXRASAM address regions don't match. You may need to review its configuration.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Warning.scg_hnf_count not valid

Fields:

SCG_HNF_COUNT unsigned int
: Number of HNF configured for the group.

SCG_REGION_NUM unsigned int
: System Cache Group region number.

event_propagation_downstream

Fields:

STATE unsigned int
: State being transmitted.

event_propagation_downstream_routing

Fields:

LINK unsigned int
: Target link.

PORT unsigned int
: Activated port.

STATE unsigned int
: State being transmitted.

event_propagation_upstream

Fields:

STATE unsigned int
: State being transmitted.

trace_add_cxra_memory_region

Fields:

END unsigned int
: End address of the memory region.

HAID unsigned int
: Memory region target HAID

START unsigned int
: Start address of the memory region.

trace_add_memory_region

Fields:

END unsigned int
: End address of the memory region.

START unsigned int
: Start address of the memory region.

TYPE enum
: Node type.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.60 ClockDivider - trace

This section describes the trace sources.

frequency

Notify when changes of frequency occur. Fields:

current_time_in_simulator_ticks unsigned int
: The current time in simulator ticks (and not clock ticks).

input_frequency_Hz unsigned int
: The input frequency for this clock source (truncated to an integer).

known_invalid_frequency bool
: The input frequency was known to be invalid and so the output frequency is too. Not all frequency sources will provide information on the validity of the frequency it advertises.

output_frequency_Hz unsigned int
: The output frequency of this clock source (truncated to an integer).

rate

Notify when the rate of a ClockDivider has been changed. Fields:

current_time_in_simulator_ticks unsigned int
: The current time in simulator ticks (and not clock ticks).

div unsigned int
: Frequency Divider

mul unsigned int
: Frequency Multiplier

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.61 D71 - trace

This section describes the trace sources.

ArchMsg.Error.IgnoredRead

Register read in reserved area. Fields:

OFFSET unsigned int
: Access offset

ArchMsg.Error.IgnoredWrite

Register write in reserved area. Fields:

DATA unsigned int
: The data written to the address

OFFSET unsigned int
: Access offset

ArchMsg.Error.ReadFromWriteOnlyRegister

Register read of write only register. Fields:

NAME string
: Register name

OFFSET unsigned int
: Access offset

ArchMsg.Error.ReadInReset

Register read during reset. Fields:

OFFSET unsigned int
: Register access offset

ArchMsg.Error.WriteInReset

Register write during reset. Fields:

OFFSET unsigned int
: Register access offset

VALUE unsigned int
: Value written.

ArchMsg.Error.WriteToReadOnlyRegister

Register write of read only register. Fields:

DATA unsigned int
: The data written to the address

NAME string
: Register name

OFFSET unsigned int
: Access offset

ArchMsg.Info.DOU_0.BackendDisabled

Backend disabled.

ArchMsg.Info.DOU_0.CmdModeNoCval

No next frame validated in command mode.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Info.DOU_0.UnsuccessfulDisplayLock

Unable to access visualisation component.

ArchMsg.Info.DOU_1.BackendDisabled

Backend disabled.

ArchMsg.Info.DOU_1.CmdModeNoCval

No next frame validated in command mode.

ArchMsg.Info.DOU_1.UnsuccessfulDisplayLock

Unable to access visualisation component.

ArchMsg.Info.GCU.ModeChange

GCU mode changed. Fields:

NEW_MODE enum
: New mode

OLD_MODE enum
: Old mode

ArchMsg.Info.LPU_0.WriteBack.NullBufferInWriteback

Received NULL buffer from INPUT_ID. Writing a solid grey color (0x808080). Fields:

INPUT_ID unsigned int
: Input block ID configured in LW_INPUT_ID0

ArchMsg.Info.LPU_1.WriteBack.NullBufferInWriteback

Received NULL buffer from INPUT_ID. Writing a solid grey color (0x808080). Fields:

INPUT_ID unsigned int
: Input block ID configured in LW_INPUT_ID0

ArchMsg.Info.RegisterRead

Register read. Fields:

OFFSET unsigned int
: Register access offset

REG_NAME string
: Name of the register being read.

VALUE unsigned int
: Value returned.

ArchMsg.Info.RegisterWrite

Register write. Fields:

OFFSET unsigned int
: Register access offset

REG_NAME string
: Name of the register being written.

VALUE unsigned int
: Value written.

ArchMsg.Info.SharedBlock.InvalidInvGammaLUTType

Invalid Inverse Gamma LUT type (ITSEL) received. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ITSEL unsigned int
: LUT Type configured in ITSEL field.

ArchMsg.Info.SharedBlock.traceFwdLUTTypeInvalid

Invalid Gamma LUT type (FTSEL) received. Fields:

FTSEL unsigned int
: LUT Type configured in FTSEL field.

ArchMsg.Warning.ADU.CVALReceivedWhileActive

FAFBC DMA Unit received CVAL event while being active. This is illegal.

ArchMsg.Warning.CU_0.InvalidSplitScenario

Invalid configuration for a Split scenario. Fields:

LSD bool
: Layer Split disabled in both scalers

LSE bool
: Layer Split enabled in both scalers

MG bool
: Merger enabled

SP bool
: Splitter enabled

ArchMsg.Warning.CU_0.Merger.InvalidInputBlockToMerger

Invalid configuration of MG_INPUT_ID. Will write a solid BLUE color while merging. Fields:

INDEX unsigned int
: Input number

INPUT_ID unsigned int
: Input block ID configured in MG_INPUT_ID

ArchMsg.Warning.CU_0.Scaler_0.DisabledScalerFetchRequest

Data fetch request received while scaler disabled. Will return a solid RED color if/when getLayer() is called.

ArchMsg.Warning.CU_0.Scaler_0.InvalidInputBlockToScaler

Invalid configuration of SC_INPUT_ID0. Writing a solid RED color. Fields:

INPUT_ID unsigned int
: Input block ID configured in SC_INPUT_ID0

ArchMsg.Warning.CU_0.Scaler_0.WrongBufferSizeToScaler

Buffer received in Scaler does not match the expected input size. Fields:

EXPCTD_H unsigned int
: Expected height

EXPCTD_W unsigned int
: Expected width

RCVD_H unsigned int
: Received height

RCVD_W unsigned int
: Received width

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.CU_0.Scaler_1.DisabledScalerFetchRequest

Data fetch request received while scaler disabled. Will return a solid RED color if/when getLayer() is called.

ArchMsg.Warning.CU_0.Scaler_1.InvalidInputBlockToScaler

Invalid configuration of SC_INPUT_ID0. Writing a solid RED color. Fields:

INPUT_ID unsigned int
: Input block ID configured in SC_INPUT_ID0

ArchMsg.Warning.CU_0.Scaler_1.WrongBufferSizeToScaler

Buffer received in Scaler does not match the expected input size. Fields:

EXPCTD_H unsigned int
: Expected height

EXPCTD_W unsigned int
: Expected width

RCVD_H unsigned int
: Received height

RCVD_W unsigned int
: Received width

ArchMsg.Warning.CU_0.Splitter.InvalidInputBlockToSplitter

Invalid configuration of SP_INPUT_ID0. Will write a solid GREEN color while splitting. Fields:

INPUT_ID unsigned int
: Input block ID configured in SP_INPUT_ID0

ArchMsg.Warning.CU_1.InvalidSplitScenario

Invalid configuration for a Split scenario. Fields:

LSD bool
: Layer Split disabled in both scalers

LSE bool
: Layer Split enabled in both scalers

MG bool
: Merger enabled

SP bool
: Splitter enabled

ArchMsg.Warning.CU_1.Merger.InvalidInputBlockToMerger

Invalid configuration of MG_INPUT_ID. Will write a solid BLUE color while merging. Fields:

INDEX unsigned int
: Input number

INPUT_ID unsigned int
: Input block ID configured in MG_INPUT_ID

ArchMsg.Warning.CU_1.Scaler_0.DisabledScalerFetchRequest

Data fetch request received while scaler disabled. Will return a solid RED color if/when getLayer() is called.

ArchMsg.Warning.CU_1.Scaler_0.InvalidInputBlockToScaler

Invalid configuration of SC_INPUT_ID0. Writing a solid RED color. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INPUT_ID unsigned int
: Input block ID configured in SC_INPUT_ID0

ArchMsg.Warning.CU_1.Scaler_0.WrongBufferSizeToScaler

Buffer received in Scaler does not match the expected input size. Fields:

EXPCTD_H unsigned int
: Expected height

EXPCTD_W unsigned int
: Expected width

RCVD_H unsigned int
: Received height

RCVD_W unsigned int
: Received width

ArchMsg.Warning.CU_1.Scaler_1.DisabledScalerFetchRequest

Data fetch request received while scaler disabled. Will return a solid RED color if/when getLayer() is called.

ArchMsg.Warning.CU_1.Scaler_1.InvalidInputBlockToScaler

Invalid configuration of SC_INPUT_ID0. Writing a solid RED color. Fields:

INPUT_ID unsigned int
: Input block ID configured in SC_INPUT_ID0

ArchMsg.Warning.CU_1.Scaler_1.WrongBufferSizeToScaler

Buffer received in Scaler does not match the expected input size. Fields:

EXPCTD_H unsigned int
: Expected height

EXPCTD_W unsigned int
: Expected width

RCVD_H unsigned int
: Received height

RCVD_W unsigned int
: Received width

ArchMsg.Warning.CU_1.Splitter.InvalidInputBlockToSplitter

Invalid configuration of SP_INPUT_ID0. Will write a solid GREEN color while splitting. Fields:

INPUT_ID unsigned int
: Input block ID configured in SP_INPUT_ID0

ArchMsg.Warning.DOU_0.IPS.SBSWithDLDisabled

Invalid configuration of Side-by-side processing. Dual-link-display disabled but Side-by-side enabled. Disabling Side-by-side processing.

ArchMsg.Warning.DOU_0.NullBufferToDisplay

Unable to create output data, displaying background colour.

ArchMsg.Warning.DOU_0.WrongVisualisationBuffer

Incorrect buffer returned by Visualisation, skipping frame output.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.DOU_1.IPS.SBSWithDLDisabled

Invalid configuration of Side-by-side processing. Dual-link-display disabled but Side-by-side enabled. Disabling Side-by-side processing.

ArchMsg.Warning.DOU_1.NullBufferToDisplay

Unable to create output data, displaying background colour.

ArchMsg.Warning.DOU_1.WrongVisualisationBuffer

Incorrect buffer returned by Visualisation, skipping frame output.

ArchMsg.Warning.GCU.ConfigValidInWrongMode

GCU received CVAL while in wrong mode (No display active). Fields:

MODE enum
: GCU mode when CVAL was received

ArchMsg.Warning.GCU.InvalidModeChangeRequest

GCU received invalid mode change request. Fields:

CUR_MODE enum
: Current mode

REQ_MODE enum
: Requested mode

ArchMsg.Warning.LPU_0.Layer_0.DisabledLayerFetchRequest

Data fetch request received while layer disabled. Will return a solid GOLD color if/when getLayer() is called.

ArchMsg.Warning.LPU_0.Layer_0.SimpleLayerYUVConfig

This is a SimpleLayer configured with YUV format. A solid GOLD color will be returned.

ArchMsg.Warning.LPU_0.Layer_0.TBUEnabledButDisonnected

TBU not connected but enabled in Layer. Passing the request to AXI port.

ArchMsg.Warning.LPU_0.Layer_0.UnsupportedInputFormat

Format is not supported for layer input. Fields:

FORMAT unsigned int
: Pixel Format

ArchMsg.Warning.LPU_0.Layer_1.DisabledLayerFetchRequest

Data fetch request received while layer disabled. Will return a solid GOLD color if/when getLayer() is called.

ArchMsg.Warning.LPU_0.Layer_1.SimpleLayerYUVConfig

This is a SimpleLayer configured with YUV format. A solid GOLD color will be returned.

ArchMsg.Warning.LPU_0.Layer_1.TBUEnabledButDisonnected

TBU not connected but enabled in Layer. Passing the request to AXI port.

ArchMsg.Warning.LPU_0.Layer_1.UnsupportedInputFormat

Format is not supported for layer input. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

FORMAT unsigned int
Pixel Format

ArchMsg.Warning.LPU_0.Layer_2.DisabledLayerFetchRequest

Data fetch request received while layer disabled. Will return a solid GOLD color if/when getLayer() is called.

ArchMsg.Warning.LPU_0.Layer_2.SimpleLayerYUVConfig

This is a SimpleLayer configured with YUV format. A solid GOLD color will be returned.

ArchMsg.Warning.LPU_0.Layer_2.TBUEnabledButDisonnected

TBU not connected but enabled in Layer. Passing the request to AXI port.

ArchMsg.Warning.LPU_0.Layer_2.UnsupportedInputFormat

Format is not supported for layer input. Fields:

FORMAT unsigned int
Pixel Format

ArchMsg.Warning.LPU_0.Layer_3.DisabledLayerFetchRequest

Data fetch request received while layer disabled. Will return a solid GOLD color if/when getLayer() is called.

ArchMsg.Warning.LPU_0.Layer_3.SimpleLayerYUVConfig

This is a SimpleLayer configured with YUV format. A solid GOLD color will be returned.

ArchMsg.Warning.LPU_0.Layer_3.TBUEnabledButDisonnected

TBU not connected but enabled in Layer. Passing the request to AXI port.

ArchMsg.Warning.LPU_0.Layer_3.UnsupportedInputFormat

Format is not supported for layer input. Fields:

FORMAT unsigned int
Pixel Format

ArchMsg.Warning.LPU_0.WriteBack.InvalidInputBlockToWriteback

Invalid configuration of LW_INPUT_ID0. Writing a solid grey color (0x404040). Fields:

INPUT_ID unsigned int
Input block ID configured in LW_INPUT_ID0

ArchMsg.Warning.LPU_0.WriteBack.TBUEnabledButDisonnected

TBU not connected but enabled in Writeback Layer. Passing the request to AXI port.

ArchMsg.Warning.LPU_0.WriteBack.UnsupportedWritebackFormat

Format not supported for writeback. Fields:

FORMAT unsigned int
Pixel Format

ArchMsg.Warning.LPU_1.Layer_0.DisabledLayerFetchRequest

Data fetch request received while layer disabled. Will return a solid GOLD color if/when getLayer() is called.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.LPU_1.Layer_0.SimpleLayerYUVConfig

This is a SimpleLayer configured with YUV format. A solid GOLD color will be returned.

ArchMsg.Warning.LPU_1.Layer_0.TBUEnabledButDisonnected

TBU not connected but enabled in Layer. Passing the request to AXI port.

ArchMsg.Warning.LPU_1.Layer_0.UnsupportedInputFormat

Format is not supported for layer input. Fields:

FORMAT `unsigned int`
: Pixel Format

ArchMsg.Warning.LPU_1.Layer_1.DisabledLayerFetchRequest

Data fetch request received while layer disabled. Will return a solid GOLD color if/when getLayer() is called.

ArchMsg.Warning.LPU_1.Layer_1.SimpleLayerYUVConfig

This is a SimpleLayer configured with YUV format. A solid GOLD color will be returned.

ArchMsg.Warning.LPU_1.Layer_1.TBUEnabledButDisonnected

TBU not connected but enabled in Layer. Passing the request to AXI port.

ArchMsg.Warning.LPU_1.Layer_1.UnsupportedInputFormat

Format is not supported for layer input. Fields:

FORMAT `unsigned int`
: Pixel Format

ArchMsg.Warning.LPU_1.Layer_2.DisabledLayerFetchRequest

Data fetch request received while layer disabled. Will return a solid GOLD color if/when getLayer() is called.

ArchMsg.Warning.LPU_1.Layer_2.SimpleLayerYUVConfig

This is a SimpleLayer configured with YUV format. A solid GOLD color will be returned.

ArchMsg.Warning.LPU_1.Layer_2.TBUEnabledButDisonnected

TBU not connected but enabled in Layer. Passing the request to AXI port.

ArchMsg.Warning.LPU_1.Layer_2.UnsupportedInputFormat

Format is not supported for layer input. Fields:

FORMAT `unsigned int`
: Pixel Format

ArchMsg.Warning.LPU_1.Layer_3.DisabledLayerFetchRequest

Data fetch request received while layer disabled. Will return a solid GOLD color if/when getLayer() is called.

ArchMsg.Warning.LPU_1.Layer_3.SimpleLayerYUVConfig

This is a SimpleLayer configured with YUV format. A solid GOLD color will be returned.

ArchMsg.Warning.LPU_1.Layer_3.TBUEnabledButDisonnected

TBU not connected but enabled in Layer. Passing the request to AXI port.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.LPU_1.Layer_3.UnsupportedInputFormat

Format is not supported for layer input. Fields:

FORMAT unsigned int
: Pixel Format

ArchMsg.Warning.LPU_1.WriteBack.InvalidInputBlockToWriteback

Invalid configuration of LW_INPUT_ID0. Writing a solid grey color (0x404040). Fields:

INPUT_ID unsigned int
: Input block ID configured in LW_INPUT_ID0

ArchMsg.Warning.LPU_1.WriteBack.TBUEnabledButDisonnected

TBU not connected but enabled in Writeback Layer. Passing the request to AXI port.

ArchMsg.Warning.LPU_1.WriteBack.UnsupportedWritebackFormat

Format not supported for writeback. Fields:

FORMAT unsigned int
: Pixel Format

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.62 DMC-500 - trace

This section describes the trace sources.

DMC_AccessInResetState

DMC Register access in reset state.

DMC_AccessToUnimplementedRegister

Accessed an unimplemented register. Fields:

ADDR unsigned int
: Address of the accessed register.

READ bool
: Transaction type is read or write.

DMC_BlockingTransactions

Blocking transaction detected. Fields:

ADDR unsigned int
: Address for the transaction.

MASTER_ID unsigned int
: Which master initiated the transaction.

NS bool
: Whether the transaction is secure or non-secure.

READ bool
: Transaction type is read or write.

DMC_CheckPermissions

Permission check performed. Fields:

ADDR unsigned int
: Transaction address.

MASTER_ID unsigned int
: Which master initiated the transaction.

NS bool
: Whether the transaction is secure or non-secure.

OUTSIDE_DEFAULT bool
: Whether the access failed due to outside default region.

READ bool
: Transaction type is read or write.

SUCCESS bool
: Whether transaction succeeded or failed.

TZ_FAIL bool
: Whether the access failed due to invalid permissions.

DMC_ReadToWriteOnlyRegister

Read to a write only register. Fields:

ADDR unsigned int
: Address of the accessed register.

DMC_RegRead

Register read. Fields:

REG_NAME string
: The name of the register read.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

REG_OFFSET unsigned int
: The address of the register read.

VALUE unsigned int
: Read Value from the register.

DMC_RegWrite

Register written. Fields:

REG_NAME string
: The name of the register updated.

REG_OFFSET unsigned int
: The address of the register updated.

UPDATED_VALUE unsigned int
: New Value in the register.

VALUE unsigned int
: Old Value in the register.

DMC_RestrictedAccessToRegister

Register access restricted and not allowed in current DMC state. Fields:

DMC_STATE unsigned int
: Architectural state of the DMC.

READ bool
: Access type is read or write.

REG_NAME string
: The name of the register updated.

REG_OFFSET unsigned int
: The address of the register accessed.

REG_RESTRICTIONS unsigned int
: Access restrictions mask of the register.

DMC_WriteToReadOnlyRegister

Write to a read only register. Fields:

ADDR unsigned int
: Address of the accessed register.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.63 DMC-520 - trace

This section describes the trace sources.

DMC520_ArchStateUpdate

Attempt to change architectural state due to command. Fields:

CMD enum
: Command received to process.

CURRENT_STATE enum
: State before the command processed.

NEW_STATE enum
: Final state after command processed.

SUCCESS bool
: Whether succeeded or failed.

DMC_AccessInResetState

DMC Register access in reset state.

DMC_AccessToUnimplementedRegister

Accessed an unimplemented register. Fields:

ADDR unsigned int
: Address of the accessed register.

READ bool
: Transaction type is read or write.

DMC_BlockingTransactions

Blocking transaction detected. Fields:

ADDR unsigned int
: Address for the transaction.

MASTER_ID unsigned int
: Which master initiated the transaction.

NS bool
: Whether the transaction is secure or non-secure.

READ bool
: Transaction type is read or write.

DMC_CheckPermissions

Permission check performed. Fields:

ADDR unsigned int
: Transaction address.

MASTER_ID unsigned int
: Which master initiated the transaction.

NS bool
: Whether the transaction is secure or non-secure.

OUTSIDE_DEFAULT bool
: Whether the access failed due to outside default region.

READ bool
: Transaction type is read or write.

SUCCESS bool
: Whether transaction succeeded or failed.

TZ_FAIL bool
: Whether the access failed due to invalid permissions.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DMC_ReadToWriteOnlyRegister

Read to a write only register. Fields:

ADDR unsigned int
: Address of the accessed register.

DMC_RegRead

Register read. Fields:

REG_NAME string
: The name of the register read.

REG_OFFSET unsigned int
: The address of the register read.

VALUE unsigned int
: Read Value from the register.

DMC_RegWrite

Register written. Fields:

REG_NAME string
: The name of the register updated.

REG_OFFSET unsigned int
: The address of the register updated.

UPDATED_VALUE unsigned int
: New Value in the register.

VALUE unsigned int
: Old Value in the register.

DMC_RestrictedAccessToRegister

Register access restricted and not allowed in current DMC state. Fields:

DMC_STATE unsigned int
: Architectural state of the DMC.

READ bool
: Access type is read or write.

REG_NAME string
: The name of the register updated.

REG_OFFSET unsigned int
: The address of the register accessed.

REG_RESTRICTIONS unsigned int
: Access restrictions mask of the register.

DMC_WriteToReadOnlyRegister

Write to a read only register. Fields:

ADDR unsigned int
: Address of the accessed register.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.64 DMC-620 - trace

This section describes the trace sources.

DMC620_ArchStateUpdate

Attempt to change architectural state due to command. Fields:

CMD enum
: Command received to process.

CURRENT_STATE enum
: State before the command processed.

NEW_STATE enum
: Final state after command processed.

SUCCESS bool
: Whether succeeded or failed.

DMC620_DetectedRASError

DMC620 RAS error detected. Fields:

RAS_ERROR enum
: Type of detected RAS error.

DMC620_Interrupt

DMC620 interrupt raised. Fields:

INTERRUPT_RAISED enum
: Type of the raised interrupt.

DMC_AccessInResetState

DMC Register access in reset state.

DMC_AccessToUnimplementedRegister

Accessed an unimplemented register. Fields:

ADDR unsigned int
: Address of the accessed register.

READ bool
: Transaction type is read or write.

DMC_BlockingTransactions

Blocking transaction detected. Fields:

ADDR unsigned int
: Address for the transaction.

MASTER_ID unsigned int
: Which master initiated the transaction.

NS bool
: Whether the transaction is secure or non-secure.

READ bool
: Transaction type is read or write.

DMC_CheckPermissions

Permission check performed. Fields:

ADDR unsigned int
: Transaction address.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MASTER_ID unsigned int
: Which master initiated the transaction.

NS bool
: Whether the transaction is secure or non-secure.

OUTSIDE_DEFAULT bool
: Whether the access failed due to outside default region.

READ bool
: Transaction type is read or write.

SUCCESS bool
: Whether transaction succeeded or failed.

TZ_FAIL bool
: Whether the access failed due to invalid permissions.

DMC_ReadToWriteOnlyRegister

Read to a write only register. Fields:

ADDR unsigned int
: Address of the accessed register.

DMC_RegRead

Register read. Fields:

REG_NAME string
: The name of the register read.

REG_OFFSET unsigned int
: The address of the register read.

VALUE unsigned int
: Read Value from the register.

DMC_RegWrite

Register written. Fields:

REG_NAME string
: The name of the register updated.

REG_OFFSET unsigned int
: The address of the register updated.

UPDATED_VALUE unsigned int
: New Value in the register.

VALUE unsigned int
: Old Value in the register.

DMC_RestrictedAccessToRegister

Register access restricted and not allowed in current DMC state. Fields:

DMC_STATE unsigned int
: Architectural state of the DMC.

READ bool
: Access type is read or write.

REG_NAME string
: The name of the register updated.

REG_OFFSET unsigned int
: The address of the register accessed.

REG_RESTRICTIONS unsigned int
: Access restrictions mask of the register.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DMC_WriteToReadOnlyRegister

Write to a read only register. Fields:

ADDR `unsigned int`
: Address of the accessed register.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.65 DP500 - trace

This section describes the trace sources.

ArchMsg.Error.MultipleScalingRequests

Multiple scaling requests in the same frame. Fields:

STAGE string
: The current processing stage.

ArchMsg.Error.MultipleWriteRequests

Multiple write requests in the same frame. Fields:

STAGE string
: The current processing stage.

ArchMsg.Info.IgnoredRead

Register read in reserved area. Fields:

NSECURE bool
: True when the access is not a secure access

OFFSET unsigned int
: Access offset

ArchMsg.Info.IgnoredWrite

Register write in reserved area. Fields:

DATA unsigned int
: The data written to the address

NSECURE bool
: True when the access is not a secure access

OFFSET unsigned int
: Access offset

ArchMsg.Info.Initialised

Initialised.

ArchMsg.Info.Processing

Processing layer. Fields:

STAGE string
: The current processing stage

ArchMsg.Info.ReadFromWriteOnlyRegister

Register read of write only register. Fields:

NSECURE bool
: True when the access is not a secure access

OFFSET unsigned int
: Access offset

ArchMsg.Info.ReadInReset

Register read during reset. Fields:

NSECURE bool
: True when the access is not a secure access

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OFFSET unsigned int
: Access offset

ArchMsg.Info.RegisterRead

Register read. Fields:

DATA unsigned int
: The data returned by the component

NSECURE bool
: True when the access is not a secure access

OFFSET unsigned int
: Register access offset

ArchMsg.Info.RegisterWrite

Register write. Fields:

DATA unsigned int
: The data written to the register

NSECURE bool
: True when the access is not a secure access

OFFSET unsigned int
: Register access offset

ArchMsg.Info.Resetting

Resetting.

ArchMsg.Info.TimerFired

Timer has fired.

ArchMsg.Info.WriteInReset

Register write during reset. Fields:

DATA unsigned int
: The data written to the address

NSECURE bool
: True when the access is not a secure access

OFFSET unsigned int
: Access offset

ArchMsg.Info.WriteToReadOnlyRegister

Register write of read only register. Fields:

DATA unsigned int
: The data written to the address

NSECURE bool
: True when the access is not a secure access

OFFSET unsigned int
: Access offset

ArchMsg.Warn.DisplayLockFailed

Couldn't acquire display lock.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.66 DP550 - trace

This section describes the trace sources.

ArchMsg.Error.IgnoredRead

Register read in reserved area. Fields:

NSECURE bool
: True when the access is not a secure access

OFFSET unsigned int
: Access offset

ArchMsg.Error.IgnoredWrite

Register write in reserved area. Fields:

DATA unsigned int
: The data written to the address

NSECURE bool
: True when the access is not a secure access

OFFSET unsigned int
: Access offset

ArchMsg.Error.ReadFromWriteOnlyRegister

Register read of write only register. Fields:

NAME string
: Register name

NSECURE bool
: True when the access is not a secure access

OFFSET unsigned int
: Access offset

ArchMsg.Error.ReadInReset

Register read during reset. Fields:

NSECURE bool
: True when the access is not a secure access

OFFSET unsigned int
: Access offset

ArchMsg.Error.UnsecureReadToSecureRegister

Non secure register read of secure register. Fields:

NAME string
: Register name

NSECURE bool
: True when the access is not a secure access

OFFSET unsigned int
: Access offset

ArchMsg.Error.UnsecureWriteoSecureRegister

Non secure register write of secure register. Fields:

DATA unsigned int
: The data written to the address

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NAME string
: Register name

NSECURE bool
: True when the access is not a secure access

OFFSET unsigned int
: Access offset

ArchMsg.Error.WriteInReset

Register write during reset. Fields:

DATA unsigned int
: The data written to the address

NSECURE bool
: True when the access is not a secure access

OFFSET unsigned int
: Access offset

ArchMsg.Error.WriteToReadOnlyRegister

Register write of read only register. Fields:

DATA unsigned int
: The data written to the address

NAME string
: Register name

NSECURE bool
: True when the access is not a secure access

OFFSET unsigned int
: Access offset

ArchMsg.Info.Initialised

Initialised.

ArchMsg.Info.Processing

Processing layer. Fields:

STAGE string
: The current processing stage

ArchMsg.Info.RegisterRead

Register read. Fields:

DATA unsigned int
: The data returned by the component

NAME string
: Register name

NSECURE bool
: True when the access is not a secure access

OFFSET unsigned int
: Register access offset

ArchMsg.Info.RegisterWrite

Register write. Fields:

DATA unsigned int
: The data written to the register

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NAME `string`
: Register name

NSECURE `bool`
: True when the access is not a secure access

OFFSET `unsigned int`
: Register access offset

ArchMsg.Info.Resetting

Resetting.

ArchMsg.Info.TimerFired

Timer has fired.

ArchMsg.Warn.DisplayLockFailed

Couldn't acquire display lock.

ArchMsg.Warning.FrameDropped

Frame tried to compose before the previous one finished. It will be dropped.

ArchMsg.Warning.clockTooHigh

The pixel clock is very high, it may affect model performance. Fields:

RATE `unsigned int`
: Refresh rate

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.67 DP650 - trace

This section describes the trace sources.

ArchMsg.Error.IgnoredRead

Register read in reserved area. Fields:

NSECURE bool
: True when the access is not a secure access

OFFSET unsigned int
: Access offset

ArchMsg.Error.IgnoredWrite

Register write in reserved area. Fields:

DATA unsigned int
: The data written to the address

NSECURE bool
: True when the access is not a secure access

OFFSET unsigned int
: Access offset

ArchMsg.Error.ReadFromWriteOnlyRegister

Register read of write only register. Fields:

NAME string
: Register name

NSECURE bool
: True when the access is not a secure access

OFFSET unsigned int
: Access offset

ArchMsg.Error.ReadInReset

Register read during reset. Fields:

NSECURE bool
: True when the access is not a secure access

OFFSET unsigned int
: Access offset

ArchMsg.Error.UnsecureReadToSecureRegister

Non secure register read of secure register. Fields:

NAME string
: Register name

NSECURE bool
: True when the access is not a secure access

OFFSET unsigned int
: Access offset

ArchMsg.Error.UnsecureWriteoSecureRegister

Non secure register write of secure register. Fields:

DATA unsigned int
: The data written to the address

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NAME string
: Register name

NSECURE bool
: True when the access is not a secure access

OFFSET unsigned int
: Access offset

ArchMsg.Error.WriteInReset

Register write during reset. Fields:

DATA unsigned int
: The data written to the address

NSECURE bool
: True when the access is not a secure access

OFFSET unsigned int
: Access offset

ArchMsg.Error.WriteToReadOnlyRegister

Register write of read only register. Fields:

DATA unsigned int
: The data written to the address

NAME string
: Register name

NSECURE bool
: True when the access is not a secure access

OFFSET unsigned int
: Access offset

ArchMsg.Info.Initialised

Initialised.

ArchMsg.Info.Processing

Processing layer. Fields:

STAGE string
: The current processing stage

ArchMsg.Info.RegisterRead

Register read. Fields:

DATA unsigned int
: The data returned by the component

NAME string
: Register name

NSECURE bool
: True when the access is not a secure access

OFFSET unsigned int
: Register access offset

ArchMsg.Info.RegisterWrite

Register write. Fields:

DATA unsigned int
: The data written to the register

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NAME string
: Register name

NSECURE bool
: True when the access is not a secure access

OFFSET unsigned int
: Register access offset

ArchMsg.Info.Resetting

Resetting.

ArchMsg.Info.TimerFired

Timer has fired.

ArchMsg.Warn.DisplayLockFailed

Couldn't acquire display lock.

ArchMsg.Warning.FrameDropped

Frame tried to compose before the previous one finished. It will be dropped.

ArchMsg.Warning.clockTooHigh

The pixel clock is very high, it may affect model performance. Fields:

RATE unsigned int
: Refresh rate

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.68 DSU - trace

This section describes the trace sources.

PSTATE_PACTIVE_OPMODE_TRANSITION

Trace pactive and Opmode mode transitions on PChannel. Fields:

DEVPACTIVE `unsigned int`
: Integer representation of DEVPACTIVE[] signals

NEW_STATE `enum`
: Operating mode after the transition

OLD_STATE `enum`
: Operating mode before the transition

PSTATE_PACTIVE_POWER_TRANSITION

Trace pactive and power mode transitions on PChannel. Fields:

DEVPACTIVE `unsigned int`
: Integer representation of DEVPACTIVE[] signals

NEW_STATE `enum`
: Power mode after the transition

OLD_STATE `enum`
: Power mode before the transition

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.69 DualClusterSystemConfigurationBlock - trace

This section describes the trace sources.

sw_trace_event

This is a trace event generate under SW control. Fields:

cluster_and_cpu_id unsigned int
: The top byte represents the cluster id, the bottom byte the cpu id.

data0 unsigned int
: SW defined data field 0.

data1 unsigned int
: SW defined data field 1.

data2 unsigned int
: SW defined data field 2.

message string
: A message from the SW.

sequence_id unsigned int
: This is a unique id per emitted trace event.

sw_trace_event2

This is a trace event generate under SW control. Fields:

cluster_and_cpu_id unsigned int
: The top byte represents the cluster id, the bottom byte the cpu id.

data0 unsigned int
: SW defined data field 0.

data1 unsigned int
: SW defined data field 1.

data2 unsigned int
: SW defined data field 2.

message string
: A message from the SW.

sequence_id unsigned int
: This is a unique id per emitted trace event.

sw_trace_event3

This is a trace event generate under SW control. Fields:

cluster_and_cpu_id unsigned int
: The top byte represents the cluster id, the bottom byte the cpu id.

data0 unsigned int
: SW defined data field 0.

data1 unsigned int
: SW defined data field 1.

data2 unsigned int
: SW defined data field 2.

message string
: A message from the SW.

sequence_id unsigned int
: This is a unique id per emitted trace event.

sw_trace_event4

This is a trace event generate under SW control. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

cluster_and_cpu_id unsigned int
: The top byte represents the cluster id, the bottom byte the cpu id.

data0 unsigned int
: SW defined data field 0.

data1 unsigned int
: SW defined data field 1.

data2 unsigned int
: SW defined data field 2.

message string
: A message from the SW.

sequence_id unsigned int
: This is a unique id per emitted trace event.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.70 GIC_400 - trace

This section describes the trace sources.

CHECKPOINT_MESSAGE

Report error messages from the checkpointing process. Fields:

message string
: Message contents.

CHECKPOINT_RESTORE_END

Checkpoint restore completed.

CHECKPOINT_RESTORE_START

Checkpoint restore about to start.

CHECKPOINT_SAVE_END

Checkpoint save completed.

CHECKPOINT_SAVE_START

Checkpoint save about to start.

debug_out

Debug trace from the VGIC model, this contains implementation detail that is unlikely to be generally useful. Fields:

message string
: The message from the VGIC

sequence_id unsigned int
: Each successive message is labeled with an increasing number

gic_log_errors_out

Errors from the VGIC model. Fields:

message string
: The message from the VGIC

sequence_id unsigned int
: Each successive message is labeled with an increasing number

gic_log_fatal_out

Fatal error from the VGIC model. Fields:

message string
: The message from the VGIC

gic_log_warnings_out

Warnings from the VGIC model. Fields:

message string
: The message from the VGIC

sequence_id unsigned int
: Each successive message is labeled with an increasing number

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

vgic_cfgsdisable

The VGIC CFGSDISABLE pin has changed its state, this (un)locks down some configuration registers. Fields:

state bool
: The state of the signal

vgic_distributor_register_access

Distributor interface accesses on the VGIC. Fields:

cpu_id unsigned int
: The CPU that is making the request

data unsigned int
: The data in the access aligned to the LSB.

is_read enum
: Is the access a read?

offset enum
: Register within interface, top bit indicates NS state

offset_in_word unsigned int
: The offset of the access in the word.

size_in_bytes unsigned int
: The size of the access in bytes.

vgic_fiq_out

The FIQ output signal to the core. Fields:

cpu_id unsigned int
: The CPU id.

state bool
: The state of the signal

vgic_hypervisor_register_access

Hypervisor interface register accesses. Fields:

cpu_id unsigned int
: The CPU that is making the request

data unsigned int
: The data in the access aligned to the LSB.

is_read enum
: Is the access a read?

offset enum
: Register within interface, top bit indicates NS state

offset_in_word unsigned int
: The offset of the access in the word.

size_in_bytes unsigned int
: The size of the access in bytes.

vgic_irq_out

The IRQ output signal to the core. Fields:

cpu_id unsigned int
: The CPU id.

state bool
: The state of the signal

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

vgic_legacy_fiq_in

The legacy FIQ input signal into the VGIC. Fields:

cpu_id unsigned int
: The CPU id. There is one legacy IRQ/FIQ per core.

state bool
: The state of the signal

vgic_legacy_irq_in

The legacy IRQ input signal into the VGIC. Fields:

cpu_id unsigned int
: The CPU id. There is one legacy IRQ/FIQ per core.

state bool
: The state of the signal

vgic_physical_register_access

Physical interface accesses on the VGIC. Fields:

cpu_id unsigned int
: The CPU that is making the request

data unsigned int
: The data in the access aligned to the LSB.

is_read enum
: Is the access a read?

offset enum
: Register within interface, top bit indicates NS state

offset_in_word unsigned int
: The offset of the access in the word.

size_in_bytes unsigned int
: The size of the access in bytes.

vgic_reset

The reset pin on the VGIC has been changed. Fields:

state bool
: The state of the SPI signal

vgic_spi

Shared Peripheral Interrupt signal changed. Fields:

id unsigned int
: The interrupt id this will generate. SPIs start at ID 32 and so you will only see ids of >= 32

state bool
: The state of the SPI signal

vgic_vfiq_out

The VFIQ output signal to the core. Fields:

cpu_id unsigned int
: The CPU id.

state bool
: The state of the signal

vgic_virq_out

The VIRQ output signal to the core. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

cpu_id unsigned int
: The CPU id.

state bool
: The state of the signal

vgic_virtual_machine_register_access

Virtual machine interface register accesses. Fields:

cpu_id unsigned int
: The CPU that is making the request

data unsigned int
: The data in the access aligned to the LSB.

is_read enum
: Is the access a read?

offset enum
: Register within interface, top bit indicates NS state

offset_in_word unsigned int
: The offset of the access in the word.

size_in_bytes unsigned int
: The size of the access in bytes.

vgic_virtual_maintenance_interrupt

The VGIC is signaling that it needs software help from the hypervisor to maintain theVGIC interfaces. Fields:

cpu_id unsigned int
: The CPU id.

state bool
: The state of the signal

vgic_wakeup_fiq

The wakeup FIQ signal. Fields:

cpu_id unsigned int
: The CPU id.

state bool
: The state of the signal

vgic_wakeup_irq

The wakeup IRQ signal. Fields:

cpu_id unsigned int
: The CPU id.

state bool
: The state of the signal

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.71 GICv2 - trace

This section describes the trace sources.

CHECKPOINT_MESSAGE

Report error messages from the checkpointing process. Fields:

message string
: Message contents.

CHECKPOINT_RESTORE_END

Checkpoint restore completed.

CHECKPOINT_RESTORE_START

Checkpoint restore about to start.

CHECKPOINT_SAVE_END

Checkpoint save completed.

CHECKPOINT_SAVE_START

Checkpoint save about to start.

debug_out

Debug trace from the VGIC model, this contains implementation detail that is unlikely to be generally useful. Fields:

message string
: The message from the VGIC

sequence_id unsigned int
: Each successive message is labeled with an increasing number

gic_log_errors_out

Errors from the VGIC model. Fields:

message string
: The message from the VGIC

sequence_id unsigned int
: Each successive message is labeled with an increasing number

gic_log_fatal_out

Fatal error from the VGIC model. Fields:

message string
: The message from the VGIC

gic_log_warnings_out

Warnings from the VGIC model. Fields:

message string
: The message from the VGIC

sequence_id unsigned int
: Each successive message is labeled with an increasing number

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

vgic_cfgsdisable

The VGIC CFGSDISABLE pin has changed its state, this (un)locks down some configuration registers. Fields:

state bool
: The state of the signal

vgic_distributor_register_access

Distributor interface accesses on the VGIC. Fields:

cpu_id unsigned int
: The CPU that is making the request

data unsigned int
: The data in the access aligned to the LSB.

is_read enum
: Is the access a read?

offset enum
: Register within interface, top bit indicates NS state

offset_in_word unsigned int
: The offset of the access in the word.

size_in_bytes unsigned int
: The size of the access in bytes.

vgic_fiq_out

The FIQ output signal to the core. Fields:

cpu_id unsigned int
: The CPU id.

state bool
: The state of the signal

vgic_hypervisor_register_access

Hypervisor interface register accesses. Fields:

cpu_id unsigned int
: The CPU that is making the request

data unsigned int
: The data in the access aligned to the LSB.

is_read enum
: Is the access a read?

offset enum
: Register within interface, top bit indicates NS state

offset_in_word unsigned int
: The offset of the access in the word.

size_in_bytes unsigned int
: The size of the access in bytes.

vgic_irq_out

The IRQ output signal to the core. Fields:

cpu_id unsigned int
: The CPU id.

state bool
: The state of the signal

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

vgic_legacy_fiq_in

The legacy FIQ input signal into the VGIC. Fields:

cpu_id unsigned int
: The CPU id. There is one legacy IRQ/FIQ per core.

state bool
: The state of the signal

vgic_legacy_irq_in

The legacy IRQ input signal into the VGIC. Fields:

cpu_id unsigned int
: The CPU id. There is one legacy IRQ/FIQ per core.

state bool
: The state of the signal

vgic_physical_register_access

Physical interface accesses on the VGIC. Fields:

cpu_id unsigned int
: The CPU that is making the request

data unsigned int
: The data in the access aligned to the LSB.

is_read enum
: Is the access a read?

offset enum
: Register within interface, top bit indicates NS state

offset_in_word unsigned int
: The offset of the access in the word.

size_in_bytes unsigned int
: The size of the access in bytes.

vgic_reset

The reset pin on the VGIC has been changed. Fields:

state bool
: The state of the SPI signal

vgic_spi

Shared Peripheral Interrupt signal changed. Fields:

id unsigned int
: The interrupt id this will generate. SPIs start at ID 32 and so you will only see ids of >= 32

state bool
: The state of the SPI signal

vgic_vfiq_out

The VFIQ output signal to the core. Fields:

cpu_id unsigned int
: The CPU id.

state bool
: The state of the signal

vgic_virq_out

The VIRQ output signal to the core. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

cpu_id unsigned int
: The CPU id.

state bool
: The state of the signal

vgic_virtual_machine_register_access

Virtual machine interface register accesses. Fields:

cpu_id unsigned int
: The CPU that is making the request

data unsigned int
: The data in the access aligned to the LSB.

is_read enum
: Is the access a read?

offset enum
: Register within interface, top bit indicates NS state

offset_in_word unsigned int
: The offset of the access in the word.

size_in_bytes unsigned int
: The size of the access in bytes.

vgic_virtual_maintenance_interrupt

The VGIC is signaling that it needs software help from the hypervisor to maintain theVGIC interfaces. Fields:

cpu_id unsigned int
: The CPU id.

state bool
: The state of the signal

vgic_wakeup_fiq

The wakeup FIQ signal. Fields:

cpu_id unsigned int
: The CPU id.

state bool
: The state of the signal

vgic_wakeup_irq

The wakeup IRQ signal. Fields:

cpu_id unsigned int
: The CPU id.

state bool
: The state of the signal

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.72 GICv3CPUInterface - trace

This section describes the trace sources.

ArchMsg.Error.GICv3_UnknownDownstreamWrite

Distributor sent a DownstreamWrite packet with unsupported format. Acknowledging and ignoring. This is a PROTOCOL ERROR, the architecture allows system behaviour to become UNPREDICATBLE from this point onwards.

ArchMsg.Warning.GICv3.Invalidv4Inv3onlyModePacket

GICv4 packet received by GICv3-only interface. Fields:

PACKET string
: The name of the invalid packet

ArchMsg.Warning.GICv3_AttemptedToDeactivateLPI

Attempt to deactivate an LPI. Fields:

INTERRUPT_ID unsigned int
: Interrupt ID that the write attempted to end.

VIRTUAL bool
: Access was to virtual interface.

ArchMsg.Warning.GICv3_AttemptedToDeactivateNonExistentInterrupt

Attempt to deactive a non-existent interrupt. Fields:

CPUID unsigned int
: CPU ID specified in end request.

INTERRUPT_ID unsigned int
: Interrupt ID that the write attempted to end.

NS bool
: Access was non-secure.

ArchMsg.Warning.GICv3_AttemptedToDeactivateSpuriousInterrupt

Attempt to deactivate a spurious interrupt. Fields:

CPUID unsigned int
: CPU ID (probably 0, but included for completeness).

GICC_REGISTER bool
: Access was to GICC register (rather than ICC register).

NS bool
: Access was non-secure.

SPURIOUS_INTERRUPT_ID unsigned int
: Interrupt ID that the write attempted to end.

VIRTUAL bool
: Access was to virtual interface.

ArchMsg.Warning.GICv3_AttemptedToEndNonExistentInterrupt

A write has attempted to end a non-existent interrupt. Fields:

CPUID unsigned int
: CPU ID specified in end request.

INTERRUPT_ID unsigned int
: Interrupt ID that the write attempted to end.

NS bool
: Access was non-secure.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VIRTUAL bool
Access was to virtual interface.

ArchMsg.Warning.GICv3_AttemptedToEndSpuriousInterrupt

A write has attempted to end a spurious interrupt. Fields:

CPUID unsigned int
CPU ID (probably 0, but included for completeness).
NS bool
Access was non-secure.
SPURIOUS_INTERRUPT_ID unsigned int
Interrupt ID that the write attempted to end.
VIRTUAL bool
Whether the access was virtual.

ArchMsg.Warning.GICv3_AttemptedToWriteDIRRegisterWhenEOImodeNotSet

Unexpected write to DIR register when EOImode not set. Fields:

CPUID unsigned int
CPU ID (probably 0, but included for completeness).
GICC_REGISTER bool
Access was to GICC register (rather than ICC register).
INTERRUPT_ID unsigned int
Interrupt ID that the write attempted to deactivate.
NS bool
Access was non-secure.
VIRTUAL bool
Access was to virtual interface.

ArchMsg.Warning.GICv3_CFGSDISABLE_unsupported

CFGSDISABLE signal has been set but doing so has no effect.

ArchMsg.Warning.GICv3_CPUInterface.GICC.MemoryMapped_ReadIgnored

Fields:

OFFSET unsigned int
Offset of address within the GICv3 module

ArchMsg.Warning.GICv3_CPUInterface.GICC.MemoryMapped_ReadReserved

Fields:

OFFSET unsigned int
Offset of address within the GICv3 module

ArchMsg.Warning.GICv3_CPUInterface.GICC.MemoryMapped_ReadWriteOnlyReg

Fields:

OFFSET unsigned int
Offset of address within the GICv3 module
REG_NAME string
Name of the register being read

ArchMsg.Warning.GICv3_CPUInterface.GICC.MemoryMapped_WriteIgnored

Fields:

OFFSET unsigned int
Offset of address within the GICv3 module

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.GICv3_CPUInterface.GICC.MemoryMapped_WriteReadOnlyReg

Fields:

OFFSET unsigned int
: Offset of address within the GICv3 module

REG_NAME string
: Name of the register being written

VALUE unsigned int
: Value written

ArchMsg.Warning.GICv3_CPUInterface.GICC.MemoryMapped_WriteReserved

Fields:

OFFSET unsigned int
: Offset of address within the GICv3 module

VALUE unsigned int
: Value written

ArchMsg.Warning.GICv3_CPUInterface.GICH.MemoryMapped_ReadIgnored

Fields:

OFFSET unsigned int
: Offset of address within the GICv3 module

ArchMsg.Warning.GICv3_CPUInterface.GICH.MemoryMapped_ReadReserved

Fields:

OFFSET unsigned int
: Offset of address within the GICv3 module

ArchMsg.Warning.GICv3_CPUInterface.GICH.MemoryMapped_ReadWriteOnlyReg

Fields:

OFFSET unsigned int
: Offset of address within the GICv3 module

REG_NAME string
: Name of the register being read

ArchMsg.Warning.GICv3_CPUInterface.GICH.MemoryMapped_WriteIgnored

Fields:

OFFSET unsigned int
: Offset of address within the GICv3 module

ArchMsg.Warning.GICv3_CPUInterface.GICH.MemoryMapped_WriteReadOnlyReg

Fields:

OFFSET unsigned int
: Offset of address within the GICv3 module

REG_NAME string
: Name of the register being written

VALUE unsigned int
: Value written

ArchMsg.Warning.GICv3_CPUInterface.GICH.MemoryMapped_WriteReserved

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OFFSET unsigned int
: Offset of address within the GICv3 module

VALUE unsigned int
: Value written

ArchMsg.Warning.GICv3_CPUInterface.GICV.MemoryMapped_ReadIgnored

Fields:

OFFSET unsigned int
: Offset of address within the GICv3 module

ArchMsg.Warning.GICv3_CPUInterface.GICV.MemoryMapped_ReadReserved

Fields:

OFFSET unsigned int
: Offset of address within the GICv3 module

ArchMsg.Warning.GICv3_CPUInterface.GICV.MemoryMapped_ReadWriteOnlyReg

Fields:

OFFSET unsigned int
: Offset of address within the GICv3 module

REG_NAME string
: Name of the register being read

ArchMsg.Warning.GICv3_CPUInterface.GICV.MemoryMapped_WriteIgnored

Fields:

OFFSET unsigned int
: Offset of address within the GICv3 module

ArchMsg.Warning.GICv3_CPUInterface.GICV.MemoryMapped_WriteReadOnlyReg

Fields:

OFFSET unsigned int
: Offset of address within the GICv3 module

REG_NAME string
: Name of the register being written

VALUE unsigned int
: Value written

ArchMsg.Warning.GICv3_CPUInterface.GICV.MemoryMapped_WriteReserved

Fields:

OFFSET unsigned int
: Offset of address within the GICv3 module

VALUE unsigned int
: Value written

ArchMsg.Warning.GICv3_DeactivateInterruptNotSent

Write to EOI/DIR register could not trigger a deactivate. Fields:

ACCESS_NS bool
: Access was non-secure.

CPUID unsigned int
: CPU ID specified in end request.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

GROUP_1 bool
: Interrupt is a group-1 interrupt.

INTERRUPT_ID unsigned int
: Interrupt ID that the write is ending.

INTERRUPT_NS bool
: Interrupt is a non-secure interrupt.

ArchMsg.Warning.GICv3_EndInterruptIDMismatch

Interrupt ID mismatch. Fields:

END_INTERRUPT_ID unsigned int
: Interrupt ID that the write is ending.

NS bool
: Access was non-secure.

SPECIFED_INTERRUPT_ID unsigned int
: Interrupt ID that was specified by the register write.

ArchMsg.Warning.GICv3_EndInterruptNotHighestPriority

An interrupt which is not the highest-priority one has been ended. Fields:

CPUID unsigned int
: CPU ID specified in end request.

INTERRUPT_ID unsigned int
: Interrupt ID that the write is ending.

LATEST_CPUID unsigned int
: CPU ID for the interrupt that was last acknowledged.

LATEST_INTERRUPT_ID unsigned int
: Interrupt ID of the interrupt that was last acknowledged.

NS bool
: Access was non-secure.

ArchMsg.Warning.GICv3_EndVirtualInterruptNotAllowed

Invalid attempt to end a virtual interrupt. Fields:

ACTIVE bool
: Matched list register (and virtual interrupt) is active.

GROUP_1 bool
: Matched list register is for a group-1 interrupt.

INTERRUPT_ID unsigned int
: Interrupt ID that the write is ending.

PRIORITY unsigned int
: Priority of the virtual interrupt.

ArchMsg.Warning.GICv3_EndVirtualInterruptNotHighestPriority

Attempt to end a virtual interrupt that is not the highest priority one. Fields:

CPUID unsigned int
: CPU ID specified in end request.

GROUP_1 bool
: Interrupt is a group-1 interrupt.

HIGHEST_PRIORITY unsigned int
: Priority of highest priority active interrupt.

INTERRUPT_ID unsigned int
: Interrupt ID that the write is ending.

INTERRUPT_PRIORITY unsigned int
: Priority level of this interrupt.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PRIORITY unsigned int
: Priority of the virtual interrupt.

ArchMsg.Warning.GICv3_ICC_InterruptIDExceedsSupportedSize

InterruptID raw value too large. Fields:

MASKEDVALUE unsigned int
: The interrupt ID which was understood

REG string
: The name of the register written with the value

VALUE unsigned int
: The written value

ArchMsg.Warning.GICv3_ICH_LR_UnpredictableHWInterruptID

ICH_LR programmed to be associated with an UNPREDICTABLE hw interrupt. Fields:

INTERRUPT_ID unsigned int
: The hardware interrupt ID set in the LR register.

IS_GICH_REG bool
: The register being programmed is GICH_LRn (rather than ICH_LRn).

LR_NUM unsigned int
: The GICH_LRn register written.

LR_VALUE unsigned int
: The value now written to the LR register.

ArchMsg.Warning.GICv3_InvalidRegAccessForLPI

Invalid register access for LPI. Fields:

REG string
: The name of the register written with the value

VALUE unsigned int
: LPI interrupt id

ArchMsg.Warning.GICv3_ReceiveClearDoesntMatchPending

Distributor attempted to clear an interrupt which isn't the pending one. Fields:

CLEAR_INTERRUPT_ID unsigned int
: Interrupt ID that the distributor is attempting to clear.

PENDING_INTERRUPT_ID unsigned int
: Interrupt ID that is pending.

ArchMsg.Warning.GICv3_ReceiveClearWhenNoInterruptPending

Attempt to clear an interrupt when no interrupt is pending. Fields:

INTERRUPT_ID unsigned int
: Interrupt ID that the distributor is attempting to clear.

ArchMsg.Warning.GICv3_ReceivedInvalidCommandFromAbove

Fields:

ACTUAL_LENGTH unsigned int
: Actual length of the data in bytes including the header.

DATA unsigned int
: The data received (including header).

EXPECTED_LENGTH unsigned int
: The length in bytes including header that was expected to be received dependent on command type and potentially the data type in the case of a data write.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

TYPE unsigned int
The command type.

ArchMsg.Warning.GICv3_SGI_RSS_unsupported_upredictable

RSS field non zero in SGI generation when the IRI does not advertise support!.

ArchMsg.Warning.GICv3_UnexpectedAck

Unexpected command received from distributor. Fields:

COMMAND enum
The command type received.

ArchMsg.Warning.GICv3_WriteIgnoredConfiguration

A register write has been ignored. Fields:

REG_NAME string
Name of register being read or written.
VALUE unsigned int
Value attempting to be written.

GICv3_CFGSDISABLE

Fields:

STATE bool
The new state of the signal.

GICv3_CPUInterface.GICC.MemoryMapped_Read

Trace read from a GICv3 memory-mapped register. Fields:

A0 unsigned int
Affinity level 0 (for banked registers)
A1 unsigned int
Affinity level 1 (for banked registers)
A2 unsigned int
Affinity level 2 (for banked registers)
A3 unsigned int
Affinity level 3 (for banked registers)
BANKED bool
Whether the access is to a banked register
NS bool
Access is non-secure
OFFSET unsigned int
Offset of address within the GICv3 module
REG_NAME string
Name of the register being read
VALUE unsigned int
Value returned

GICv3_CPUInterface.GICC.MemoryMapped_Read64

Trace read from a GICv3 memory-mapped register. Fields:

A0 unsigned int
Affinity level 0 (for banked registers)
A1 unsigned int
Affinity level 1 (for banked registers)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

A2 unsigned int
: Affinity level 2 (for banked registers)

A3 unsigned int
: Affinity level 3 (for banked registers)

BANKED bool
: Whether the access is to a banked register

NS bool
: Access is non-secure

OFFSET unsigned int
: Offset of address within the GICv3 module

REG_NAME string
: Name of the register being read

VALUE unsigned int
: Value returned

GICv3_CPUInterface.GICC.MemoryMapped_Write

Trace write to a GICv3 memory-mapped register. Fields:

A0 unsigned int
: Affinity level 0 (for banked registers)

A1 unsigned int
: Affinity level 1 (for banked registers)

A2 unsigned int
: Affinity level 2 (for banked registers)

A3 unsigned int
: Affinity level 3 (for banked registers)

BANKED bool
: Whether the access is to a banked register

NS bool
: Access is non-secure

OFFSET unsigned int
: Offset of address within the GICv3 module

REG_NAME string
: Name of the register being written

UPDATED_VALUE unsigned int
: New value read back from the register; for write-only registers this will be equal to VALUE.

VALUE unsigned int
: Value written

GICv3_CPUInterface.GICC.MemoryMapped_Write64

Trace write to a GICv3 memory-mapped register. Fields:

A0 unsigned int
: Affinity level 0 (for banked registers)

A1 unsigned int
: Affinity level 1 (for banked registers)

A2 unsigned int
: Affinity level 2 (for banked registers)

A3 unsigned int
: Affinity level 3 (for banked registers)

BANKED bool
: Whether the access is to a banked register

NS bool
: Access is non-secure

OFFSET unsigned int
: Offset of address within the GICv3 module

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

REG_NAME string
: Name of the register being written

UPDATED_VALUE unsigned int
: New value read back from the register; for write-only registers this will be equal to VALUE.

VALUE unsigned int
: Value written

GICv3_CPUInterface.GICC.MemoryMapped_WritePort

Trace write to a GICv3 memory-mapped register acting as imp def port. Fields:

FIRST_BYTE_IN_BURST unsigned int
: First byte written in the burst

NS bool
: Access is non-secure

OFFSET unsigned int
: Offset of address within the GICv3 module

REG_NAME string
: Name of the register being written

GICv3_CPUInterface.GICH.MemoryMapped_Read

Trace read from a GICv3 memory-mapped register. Fields:

A0 unsigned int
: Affinity level 0 (for banked registers)

A1 unsigned int
: Affinity level 1 (for banked registers)

A2 unsigned int
: Affinity level 2 (for banked registers)

A3 unsigned int
: Affinity level 3 (for banked registers)

BANKED bool
: Whether the access is to a banked register

NS bool
: Access is non-secure

OFFSET unsigned int
: Offset of address within the GICv3 module

REG_NAME string
: Name of the register being read

VALUE unsigned int
: Value returned

GICv3_CPUInterface.GICH.MemoryMapped_Read64

Trace read from a GICv3 memory-mapped register. Fields:

A0 unsigned int
: Affinity level 0 (for banked registers)

A1 unsigned int
: Affinity level 1 (for banked registers)

A2 unsigned int
: Affinity level 2 (for banked registers)

A3 unsigned int
: Affinity level 3 (for banked registers)

BANKED bool
: Whether the access is to a banked register

NS bool
: Access is non-secure

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OFFSET unsigned int
: Offset of address within the GICv3 module

REG_NAME string
: Name of the register being read

VALUE unsigned int
: Value returned

GICv3_CPUInterface.GICH.MemoryMapped_Write

Trace write to a GICv3 memory-mapped register. Fields:

A0 unsigned int
: Affinity level 0 (for banked registers)

A1 unsigned int
: Affinity level 1 (for banked registers)

A2 unsigned int
: Affinity level 2 (for banked registers)

A3 unsigned int
: Affinity level 3 (for banked registers)

BANKED bool
: Whether the access is to a banked register

NS bool
: Access is non-secure

OFFSET unsigned int
: Offset of address within the GICv3 module

REG_NAME string
: Name of the register being written

UPDATED_VALUE unsigned int
: New value read back from the register; for write-only registers this will be equal to VALUE.

VALUE unsigned int
: Value written

GICv3_CPUInterface.GICH.MemoryMapped_Write64

Trace write to a GICv3 memory-mapped register. Fields:

A0 unsigned int
: Affinity level 0 (for banked registers)

A1 unsigned int
: Affinity level 1 (for banked registers)

A2 unsigned int
: Affinity level 2 (for banked registers)

A3 unsigned int
: Affinity level 3 (for banked registers)

BANKED bool
: Whether the access is to a banked register

NS bool
: Access is non-secure

OFFSET unsigned int
: Offset of address within the GICv3 module

REG_NAME string
: Name of the register being written

UPDATED_VALUE unsigned int
: New value read back from the register; for write-only registers this will be equal to VALUE.

VALUE unsigned int
: Value written

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

GICv3_CPUInterface.GICH.MemoryMapped_WritePort

Trace write to a GICv3 memory-mapped register acting as imp def port. Fields:

FIRST_BYTE_IN_BURST unsigned int
: First byte written in the burst

NS bool
: Access is non-secure

OFFSET unsigned int
: Offset of address within the GICv3 module

REG_NAME string
: Name of the register being written

GICv3_CPUInterface.GICV.MemoryMapped_Read

Trace read from a GICv3 memory-mapped register. Fields:

A0 unsigned int
: Affinity level 0 (for banked registers)

A1 unsigned int
: Affinity level 1 (for banked registers)

A2 unsigned int
: Affinity level 2 (for banked registers)

A3 unsigned int
: Affinity level 3 (for banked registers)

BANKED bool
: Whether the access is to a banked register

NS bool
: Access is non-secure

OFFSET unsigned int
: Offset of address within the GICv3 module

REG_NAME string
: Name of the register being read

VALUE unsigned int
: Value returned

GICv3_CPUInterface.GICV.MemoryMapped_Read64

Trace read from a GICv3 memory-mapped register. Fields:

A0 unsigned int
: Affinity level 0 (for banked registers)

A1 unsigned int
: Affinity level 1 (for banked registers)

A2 unsigned int
: Affinity level 2 (for banked registers)

A3 unsigned int
: Affinity level 3 (for banked registers)

BANKED bool
: Whether the access is to a banked register

NS bool
: Access is non-secure

OFFSET unsigned int
: Offset of address within the GICv3 module

REG_NAME string
: Name of the register being read

VALUE unsigned int
: Value returned

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

GICv3_CPUInterface.GICV.MemoryMapped_Write

Trace write to a GICv3 memory-mapped register. Fields:

A0 unsigned int
: Affinity level 0 (for banked registers)

A1 unsigned int
: Affinity level 1 (for banked registers)

A2 unsigned int
: Affinity level 2 (for banked registers)

A3 unsigned int
: Affinity level 3 (for banked registers)

BANKED bool
: Whether the access is to a banked register

NS bool
: Access is non-secure

OFFSET unsigned int
: Offset of address within the GICv3 module

REG_NAME string
: Name of the register being written

UPDATED_VALUE unsigned int
: New value read back from the register; for write-only registers this will be equal to VALUE.

VALUE unsigned int
: Value written

GICv3_CPUInterface.GICV.MemoryMapped_Write64

Trace write to a GICv3 memory-mapped register. Fields:

A0 unsigned int
: Affinity level 0 (for banked registers)

A1 unsigned int
: Affinity level 1 (for banked registers)

A2 unsigned int
: Affinity level 2 (for banked registers)

A3 unsigned int
: Affinity level 3 (for banked registers)

BANKED bool
: Whether the access is to a banked register

NS bool
: Access is non-secure

OFFSET unsigned int
: Offset of address within the GICv3 module

REG_NAME string
: Name of the register being written

UPDATED_VALUE unsigned int
: New value read back from the register; for write-only registers this will be equal to VALUE.

VALUE unsigned int
: Value written

GICv3_CPUInterface.GICV.MemoryMapped_WritePort

Trace write to a GICv3 memory-mapped register acting as imp def port. Fields:

FIRST_BYTE_IN_BURST unsigned int
: First byte written in the burst

NS bool
: Access is non-secure

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OFFSET unsigned int
: Offset of address within the GICv3 module

REG_NAME string
: Name of the register being written

GICv3_CPUInterface_Comms_Receive

GICv3 internal communications packet that has been received and is traveling towards the CPU. Fields:

COMMAND enum
: The command type for this packet

DATA unsigned int
: The data (including header) contained within this packet

DATA_LENGTH unsigned int
: Number of bytes in the data (including header)

GICv3_CPUInterface_Comms_Send

GICv3 internal communications packet that is being sent towards the top-level. Fields:

COMMAND enum
: The command type for this packet

DATA unsigned int
: The data (including header) contained within this packet

DATA_LENGTH unsigned int
: Number of bytes in the data (including header)

GICv3_CreateInterruptAPRWrite

An active interrupt has been created by a write to APR. Fields:

GROUP_1 bool
: Whether the interrupt is created as group 1.

NS_REG bool
: Whether the access was through GICC_NSAPR.

PRIORITY unsigned int
: Priority of the interrupt being created.

GICv3_DeactivateInterrupt

Trace writes to EOI/DIR registers that deactivate an interrupt. Fields:

ACCESS_NS bool
: Access was non-secure.

CPUID unsigned int
: CPU ID specified in end request.

GROUP_1 bool
: Interrupt is a group-1 interrupt.

INTERRUPT_ID unsigned int
: Interrupt ID that the write is ending.

INTERRUPT_NS bool
: Interrupt is a non-secure interrupt.

GICv3_EndInterrupt

Trace writes to GICC_EOIR or GICC_AEOIR that end an interrupt. Fields:

CPUID unsigned int
: CPU ID specified in end request.

DEACTIVATE bool
: Interrupt is deactivated (rather than requiring a separate DIR request); will never be true for LPIs.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

GROUP_1 bool
: Interrupt is a group-1 interrupt.

INTERRUPT_ID unsigned int
: Interrupt ID that the write is ending.

NS bool
: Access was non-secure.

GICv3_EndInterruptAPRWrite

An interrupt has been deactivated because of a write to the priority register. Fields:

INTERRUPT_ID unsigned int
: Interrupt ID that the write is ending.

NS_REG bool
: Whether the access was through GICC_NSAPR.

PRIORITY unsigned int
: Priority of the interrupt being deactivated.

GICv3_ICH_MISR_Changed

Change of ICH_MISR register. Fields:

VALUE unsigned int
: The new value for ICH_MISR.

GICv3_IRQBypassSignalIn

An interrupt bypass signal has changed state. Fields:

IS_FIQ bool
: Whether the signal in question is the FIQ signal (rather than the IRQ).

VALUE bool
: New value for the signal (i.e. whether it is set).

GICv3_IRQLegacySignalOut

An output legacy interrupt signal has changed state. Fields:

IS_FIQ bool
: Whether the signal in question is the FIQ signal (rather than the IRQ).

VALUE bool
: New value for the signal (i.e. whether it is set).

GICv3_IRQSignalOut

An interrupt signal has changed state. Fields:

IS_FIQ bool
: Whether the signal in question is the FIQ signal (rather than the IRQ).

VALUE bool
: New value for the signal (i.e. whether it is set).

VIRTUAL bool
: Whether we're changing the virtual signal.

GICv3_InterfaceQuiesced

Event fires when the CPU interface is quiesced (Quiesce Acknowledge command sent to distributor).

GICv3_MaintenanceInterruptChanged

Change of state of maintenance interrupt. Fields:

STATE bool
: Whether the maintenance interrupt was set or cleared.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

GICv3_RegUpdated32

Fields:

REG_NAME `string`
: The name of the register being updated.

VALUE `unsigned int`
: The updated value.

GICv3_RegUpdated64

Fields:

REG_NAME `string`
: The name of the register being updated.

VALUE `unsigned int`
: The updated value.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.73 GICv3CPUInterfaceDecoder - trace

This section describes the trace sources.

GICV3_DECODER_DECODE

Trace accesses through the GICv3 decoder (not including pass-through). Fields:

ADDR `unsigned int`
: Address of the access.

ADDR_OUTPUT_OFFSET `unsigned int`
: The output offset for this transaction.

FORCE_SECURE `bool`
: Whether this transaction is using the input port that forces transaction attributes to secure.

OUTPUT_PORT_NAME `string`
: The name of the output port that we're using

READnWRITE `bool`
: Whether this transaction is a read (rather than a write).

GICV3_DECODER_PASSTHRU

Trace accesses through the GICv3 decoder that are being passed straight-through. Fields:

ADDR `unsigned int`
: Address of the access.

READnWRITE `bool`
: Whether this transaction is a read (rather than a write).

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.74 GICv3CommsLogger - trace

This section describes the trace sources.

ACTIVATE

The CPU interface sends an Activate command when acknowledging an interrupt. Fields:

DIRECTION enum
: Direction of the command.

INTID unsigned int
: Interrupt number

V enum
: The Activate corresponds to a Set or VSet command.

ACTIVATE_ACKNOWLEDGE

The Redistributor sends an Activate Acknowledge response to confirm receipt of an Activate command. Fields:

DIRECTION enum
: Direction of the command.

V enum
: The Activate Acknowledge corresponds to a Set or VSet command.

CLEAR

The Clear command clears the specified pending interrupt. Fields:

DIRECTION enum
: Direction of the command.

INTID unsigned int
: Interrupt number

CLEAR_ACKNOWLEDGE

Clear Acknowledge response to acknowledge the receipt of a Clear or VClear command. Fields:

DIRECTION enum
: Direction of the command.

V enum
: The Clear Acknowlegde corresponds to a Clear or VClear command.

DEACTIVATE

The Deactivate command deactivates an interrupt, provided the initiating Exception level and Security state can access the interrupt group to which the INTID belongs. Fields:

DIRECTION enum
: Direction of the command.

GROUP0 bool
: 1=Group 0 interrupts can be modified

GROUP1_NS bool
: 1=Non-secure group 1 interrupts can be modified

GROUP1_S bool
: 1=Secure group 1 interrupts can be modified

INTID unsigned int
: Interrupt number

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DEACTIVATE_ACKNOWLEDGE

The Redistributor sends a Deactivate Acknowledge response to confirm receipt of a Deactivate command. Fields:

DIRECTION enum
: Direction of the command.

DOWNSTREAM_CONTROL_ACKNOWLEDGE

Downstream Control Acknowledge response to confirm receipt of a Downstream Control command. Fields:

DIRECTION enum
: Direction of the command.

PL enum
: Size of pINTID

VL enum
: Size of vINTID

DOWNSTREAM_CONTROL_IMP_DEF

Implementation defined command send to CPU interface. Fields:

DIRECTION enum
: Direction of the command.

DOWNSTREAM_CONTROL_SETTINGS

Communicates settings to CPU interface. Fields:

DIRECTION enum
: Direction of the command.

DS bool
: Disable Security. Indicates the value of GICD_CTLR.DS

PL enum
: Size of pINTID

VL enum
: Size of vINTID

GENERATE_SGI

The CPU interface sends a Generate SGI command to the Redistributor to generate an SGI. Fields:

A1 unsigned int
: Affinity level 1 values used for generating the set of target PEs

A2 unsigned int
: Affinity level 2 values used for generating the set of target PEs

A3 unsigned int
: Affinity level 3 values used for generating the set of target PEs

A3V bool
: Command includes an A3 field

DIRECTION enum
: Direction of the command.

IRM bool
: Interrupt Routing Mode used

NS bool
: 0=command originates from a Secure Execution state. 1=command originates from a Non-secure Execution state

SGInum unsigned int
: The INTID of the SGI to be generated

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SGT enum
: Register access that caused the Generate SGI comman

Target_List unsigned int
: The group of target PEs defined by the routing mode

GENERATE_SGI_ACKNOWLEDGE

The Redistributor sends a Generate SGI Acknowledge response to confirm that is has received a Generate SGIcommand from the CPU interface. Fields:

DIRECTION enum
: Direction of the command.

QUIESCE

The Redistributor sends a Quiesce command to request that the CPU interface enters the quiescent state. Fields:

DIRECTION enum
: Direction of the command.

QUIESCE_ACKNOWLEDGE

Quiesce Acknowledge response to confirm receipt of a Quiesce command, and to confirm that it is quiescent. Fields:

DIRECTION enum
: Direction of the command.

RELEASE

The CPU interface logic sends a Release response when it cannot handle a particular interrupt. Fields:

DIRECTION enum
: Direction of the command.

INTID unsigned int
: Interrupt number

V enum
: The Release corresponds to a Set or VSet command.

SET

The Set command sets the highest priority pending interrupt for a PE. Fields:

DIRECTION enum
: Direction of the command.

GROUP bool
: The interrupt group, as indicated by the corresponding GICD_IGROUPR<n>.Group status bit.

INTID unsigned int
: Interrupt number

MOD bool
: The value of the GICD_IGRPMODR<n>.Group status bit for the interrupt

PRIORITY unsigned int
: The actual priority of the interrupt, that is, the Secure, unshifted view

UPSTREAM_CONTROL_ACKNOWLEDGE

The Redistributor sends an Upstream Control Acknowledge response to confirm receipt of an Upstream Control command. Fields:

DIRECTION enum
: Direction of the command.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

UPSTREAM_CONTROL_PHYSICAL_ENABLE

Communicates physical CPU interface enable to the Redistributor. Fields:

DIRECTION enum
: Direction of the command.

GROUP0 bool
: The value of ICC_IGRPEN0_EL1.Enable

GROUP1_NS bool
: The value of the Non-secure copy of ICC_IGRPEN1_EL1.Enable

GROUP1_S bool
: The value of the Secure copy of ICC_IGRPEN1_EL1.Enable

UPSTREAM_CONTROL_PHYSICAL_PRIORITY

Communicates Priority Mask to the Redistributor. Fields:

DIRECTION enum
: Direction of the command.

PMR unsigned int
: The value of PMR

UPSTREAM_CONTROL_VIRTUAL_ENABLE

Communicates virtual CPU interface enable to the Redistributor. Fields:

DIRECTION enum
: Direction of the command.

GROUP0 bool
: The value of ICH_VMCR_EL2.VENG0

GROUP1 bool
: The value of ICH_VMCR_EL2.VENG1

VCLEAR

The VClear command resets the highest priority pending virtual interrupt. Fields:

DIRECTION enum
: Direction of the command.

VINTID unsigned int
: Interrupt number

VSET

The VSet command sets the highest priority pending interrupt for a PE. Fields:

DIRECTION enum
: Direction of the command.

GROUP bool
: The interrupt group

PRIORITY unsigned int
: The actual priority of the interrupt, that is, the Secure, unshifted view

VINTID unsigned int
: Interrupt number

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.75 GICv3CommsPVBus - trace

This section describes the trace sources.

ArchMsg.Warning.GICv3CommsPVBus_PacketTooBig

Fields:

DATA_0 unsigned int
: First byte of the packet (normally data type).

INDEX unsigned int
: Index of GICv3Comms connection that the message should have been sent to (based on AXI master ID).

LENGTH unsigned int
: Number bytes in data packet.

ArchMsg.Warning.GICv3CommsPVBus_UnknownDestination

Fields:

DATA unsigned int
: The message data.

DATA_0 unsigned int
: First byte of the packet (normally data type).

INDEX unsigned int
: Index of GICv3Comms connection that the message should have been sent to (based on AXI master ID).

LENGTH unsigned int
: Number bytes in data packet.

GICv3CommsPVBus_FailedMessage

Processed message from connected GICv3Comms port and sent out, but PVBus transaction failed.
Fields:

DATA unsigned int
: The message data.

INDEX unsigned int
: Index of GICv3Comms connection that message came in on.

LENGTH unsigned int
: Message length in bytes.

GICv3CommsPVBus_ProcessedMessage

Processed message from connected GICv3Comms port. Fields:

DATA unsigned int
: The message data.

INDEX unsigned int
: Index of GICv3Comms connection that message came in on.

LENGTH unsigned int
: Message length in bytes.

GICv3CommsPVBus_ReceiveAXIMasterID

Fields:

INDEX unsigned int
: Index of GICv3Comms connection that message came in on.

MASTER_ID unsigned int
: Master ID value recieved.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

GICv3CommsPVBus_SentViaGICv3Comms

Packet received via PVBus and sent via GICv3 Comms to connected distributor/CPU. Fields:

DATA `unsigned int`
: The message data.

INDEX `unsigned int`
: Index of GICv3Comms connection that the message should have been sent to (based on AXI master ID).

LENGTH `unsigned int`
: Number bytes in data packet.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.76 GICv3Distributor - trace

This section describes the trace sources.

ArchMsg.Info.GICv3_DroppedInternalPacket

Fields:

INTERFACE unsigned int
: Index of the interface

ArchMsg.Warning.GICv3_CFGSDISABLE_unsupported

CFGSDISABLE signal has been set but doing so has no effect.

ArchMsg.Warning.GICv3_Distributor.MemoryMapped_ReadIgnored

Fields:

OFFSET unsigned int
: Offset of address within the GICv3 module

ArchMsg.Warning.GICv3_Distributor.MemoryMapped_ReadReserved

Fields:

OFFSET unsigned int
: Offset of address within the GICv3 module

ArchMsg.Warning.GICv3_Distributor.MemoryMapped_ReadWriteOnlyReg

Fields:

OFFSET unsigned int
: Offset of address within the GICv3 module

REG_NAME string
: Name of the register being read

ArchMsg.Warning.GICv3_Distributor.MemoryMapped_WriteIgnored

Fields:

OFFSET unsigned int
: Offset of address within the GICv3 module

ArchMsg.Warning.GICv3_Distributor.MemoryMapped_WriteReadOnlyReg

Fields:

OFFSET unsigned int
: Offset of address within the GICv3 module

REG_NAME string
: Name of the register being written

VALUE unsigned int
: Value written

ArchMsg.Warning.GICv3_Distributor.MemoryMapped_WriteReserved

Fields:

OFFSET unsigned int
: Offset of address within the GICv3 module

VALUE unsigned int
: Value written

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.GICv3_DroppedInternalPacket

Fields:

INTERFACE unsigned int
: Index of the interface

ArchMsg.Warning.GICv3_MemoryMapped_AccessInIncorrectAREState

Fields:

ARE_NS bool
: Value of GICD_CTLR.ARE_NS.

ARE_S bool
: Value of GICD_CTLR.ARE_S.

NS bool
: Access is non-secure.

READnWRITE bool
: Access is a read (rather than a write).

REG_NAME string
: Name of the register being accessed

ArchMsg.Warning.GICv3_MemoryMapped_ChangeUnimplementedSPI

Write attempted to change the configuration for an unimplemented SPI.

ArchMsg.Warning.GICv3_MemoryMapped_IRouteR_RoutingModeWrittenWhenRazWi

Fields:

REG_NAME string
: Name of the register being accessed.

ArchMsg.Warning.GICv3_MemoryMapped_IgnoredPendingChange

Fields:

INTERRUPT_ID unsigned int
: The interrupt ID specified in the register access.

IS_PENDING bool
: The current pending state of the interrupt (and what the access attempted to set it to).

REG_NAME string
: Name of the register being accessed.

ArchMsg.Warning.GICv3_MemoryMapped_NSAccessToSecureInterruptIgnored

Fields:

INTERRUPT_ID unsigned int
: The interrupt ID specified in the register access.

REG_NAME string
: Name of the register being accessed.

ArchMsg.Warning.GICv3_MemoryMapped_NSWriteToSecureInterrupt

Fields:

INTERRUPT_ID unsigned int
: The interrupt ID specified in the register access.

READnWRITE bool
: Access is a read (rather than a write).

REG_NAME string
: Name of the register being accessed.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.GICv3_MemoryMapped_SGIRWrittenUsingReservedValue

Fields:

REG_NAME string
: Name of the register being written.

VALUE unsigned int
: Value being written to the register.

ArchMsg.Warning.GICv3_MemoryMapped_SGISendToZeroTargets

Fields:

REG_NAME string
: Name of the register being written.

VALUE unsigned int
: Value being written to the register.

ArchMsg.Warning.GICv3_MemoryMapped_SGIWrongGroup

Fields:

CPU_NS bool
: The destination CPU is configured to accept non-secure SGIs.

REG_NAME string
: Name of the register being written.

SGI_ID unsigned int
: Interrupt ID for the SGI which is attempting to be sent.

SOURCE_NO unsigned int
: GICv2 processor number for the source CPU.

TARGET_NO unsigned int
: GICv2 processor number for the destination CPU.

VALUE unsigned int
: Value being written to the register.

ArchMsg.Warning.GICv3_MemoryMapped_SPISpecifiedOutOfRange

Fields:

INTERRUPT_ID unsigned int
: The interrupt ID specified in the register access.

READnWRITE bool
: The access in question is a read rather than a write

REG_NAME string
: Name of the register being accessed.

ArchMsg.Warning.GICv3_MemoryMapped_SecureSPIRegWrittenWhenSecurityDisabled

Fields:

INTERRUPT_ID unsigned int
: The SPI attempting to be set/cleared.

REG_NAME string
: Name of the register being accessed.

ArchMsg.Warning.GICv3_ReceivedInvalidCommandFromBelow

Fields:

ACTUAL_LENGTH unsigned int
: Actual length of the data in bytes including the header.

DATA unsigned int
: The data received (including header).

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

EXPECTED_LENGTH unsigned int
: The length in bytes including header that was expected to be received dependent on command type and potentially the data type in the case of a data write.

REDISTRIBUTOR_INDEX unsigned int
: Index for the downstream port that this command was received on.

TYPE unsigned int
: The command type.

ArchMsg.Warning.GICv3_SGIDiscardedDueToInvalidTarget

Fields:

A1 unsigned int
: Affinity level 1.

A2 unsigned int
: Affinity level 2.

A3 unsigned int
: Affinity level 3.

INTERRUPT_ID unsigned int
: The ID of the SGI.

TARGET_LIST unsigned int
: Bit vector of targets.

ArchMsg.Warning.GICv3_SPI_OUT_OF_RANGE_ONLINE

Fields:

END_SPI_ID unsigned int
: End of SPI ID assigned to this distributor for multichip operation.

SPI_PIN unsigned int
: The interrupt ID for the SPI from wire.

START_SPI_ID unsigned int
: Beginning of SPI ID assigned to this distributor for multichip operation.

GICv3_CFGSDISABLE

Fields:

STATE bool
: The new state of the signal.

GICv3_Distributor.MemoryMapped_Read

Trace read from a GICv3 memory-mapped register. Fields:

A0 unsigned int
: Affinity level 0 (for banked registers)

A1 unsigned int
: Affinity level 1 (for banked registers)

A2 unsigned int
: Affinity level 2 (for banked registers)

A3 unsigned int
: Affinity level 3 (for banked registers)

BANKED bool
: Whether the access is to a banked register

NS bool
: Access is non-secure

OFFSET unsigned int
: Offset of address within the GICv3 module

REG_NAME string
: Name of the register being read

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VALUE unsigned int
Value returned

GICv3_Distributor.MemoryMapped_Read64

Trace read from a GICv3 memory-mapped register. Fields:

A0 unsigned int
Affinity level 0 (for banked registers)
A1 unsigned int
Affinity level 1 (for banked registers)
A2 unsigned int
Affinity level 2 (for banked registers)
A3 unsigned int
Affinity level 3 (for banked registers)
BANKED bool
Whether the access is to a banked register
NS bool
Access is non-secure
OFFSET unsigned int
Offset of address within the GICv3 module
REG_NAME string
Name of the register being read
VALUE unsigned int
Value returned

GICv3_Distributor.MemoryMapped_Write

Trace write to a GICv3 memory-mapped register. Fields:

A0 unsigned int
Affinity level 0 (for banked registers)
A1 unsigned int
Affinity level 1 (for banked registers)
A2 unsigned int
Affinity level 2 (for banked registers)
A3 unsigned int
Affinity level 3 (for banked registers)
BANKED bool
Whether the access is to a banked register
NS bool
Access is non-secure
OFFSET unsigned int
Offset of address within the GICv3 module
REG_NAME string
Name of the register being written
UPDATED_VALUE unsigned int
New value read back from the register; for write-only registers this will be equal to VALUE.
VALUE unsigned int
Value written

GICv3_Distributor.MemoryMapped_Write64

Trace write to a GICv3 memory-mapped register. Fields:

A0 unsigned int
Affinity level 0 (for banked registers)
A1 unsigned int
Affinity level 1 (for banked registers)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

A2 unsigned int
: Affinity level 2 (for banked registers)

A3 unsigned int
: Affinity level 3 (for banked registers)

BANKED bool
: Whether the access is to a banked register

NS bool
: Access is non-secure

OFFSET unsigned int
: Offset of address within the GICv3 module

REG_NAME string
: Name of the register being written

UPDATED_VALUE unsigned int
: New value read back from the register; for write-only registers this will be equal to VALUE.

VALUE unsigned int
: Value written

GICv3_Distributor.MemoryMapped_WritePort

Trace write to a GICv3 memory-mapped register acting as imp def port. Fields:

FIRST_BYTE_IN_BURST unsigned int
: First byte written in the burst

NS bool
: Access is non-secure

OFFSET unsigned int
: Offset of address within the GICv3 module

REG_NAME string
: Name of the register being written

GICv3_Distributor_Comms_Receive

GICv3 internal communications packet that has been received and is traveling towards the top-level. Fields:

COMMAND enum
: The command type for this packet

DATA unsigned int
: The data (including header) contained within this packet

DATA_LENGTH unsigned int
: Number of bytes in the data (including header)

INDEX unsigned int
: Index for the redistributor at the lower level from which this packet is being sent. Not applicable for sends from CPU interface to RD0.

GICv3_Distributor_Comms_Send

GICv3 internal communications packet that is being sent towards the CPU. Fields:

COMMAND enum
: The command type for this packet

DATA unsigned int
: The data (including header) contained within this packet

DATA_LENGTH unsigned int
: Number of bytes in the data (including header)

INDEX unsigned int
: Index for the redistributor at the lower level from which this packet is being sent. Not applicable for sends from CPU interface to RD0.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

GICv3_InterruptSetPendingButGroupNotEnabled

Fields:

GROUP1 bool
: Whether the interrupt is group 1.

INTERRUPT_ID unsigned int
: The interrupt ID that was set pending.

NS bool
: Whether the interrupt is non-secure.

GICv3_InterruptSetPendingButNoDestinations

Fields:

INTERRUPT_ID unsigned int
: The interrupt ID that was set pending.

GICv3_InterruptSetReservedGroup

Fields:

INTERRUPT_ID unsigned int
: The interrupt ID that was set pending.

GICv3_Multichip_PROPBASEUpdateFromRd

Fields:

PROC_NUM unsigned int
: ID of this redistributor which consists of chip id and core id

PROPBASE unsigned int
: The value of PROPBASE written at the Redistributor.

GICv3_RegUpdated32

Fields:

REG_NAME string
: The name of the register being updated.

VALUE unsigned int
: The updated value.

GICv3_RegUpdated64

Fields:

REG_NAME string
: The name of the register being updated.

VALUE unsigned int
: The updated value.

GICv3_SPISignalChanged

Fields:

INTERRUPT_ID unsigned int
: The interrupt ID for the SPI.

STATE bool
: The new state of the signal.

GICv3_TraceWakeRequest

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

INTERFACE `unsigned int`
: Index of the interface

SETnCLEAR `bool`
: Whether the signal is set(true) or cleared (false)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.77 GICv3IRI - trace

This section describes the trace sources.

CHECKPOINT_MESSAGE

Report error messages from the checkpointing process. Fields:

message string
: Message contents.

CHECKPOINT_RESTORE_END

Checkpoint restore completed.

CHECKPOINT_RESTORE_START

Checkpoint restore about to start.

CHECKPOINT_SAVE_END

Checkpoint save completed.

CHECKPOINT_SAVE_START

Checkpoint save about to start.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.78 GICv3InterruptTranslationService - trace

This section describes the trace sources.

ArchMsg.Error.GICv3_GITS_BASER_NotProvisionedAtEnable

Fields:

BASER_INDEX unsigned int
: Index of the BASER register that should have been provisioned.

BASER_TYPE enum
: The type of the BASER register that needs to have been provisioned.

ArchMsg.Warning.GICv3_CBASERUnpredAddress

Fields:

VALUE unsigned int
: Value attempting to be written.

ArchMsg.Warning.GICv3_CFGSDISABLE_unsupported

CFGSDISABLE signal has been set but doing so has no effect.

ArchMsg.Warning.GICv3_CWRITER_Write

Fields:

OFFSET unsigned int
: Offset written to CWRITE register.

ArchMsg.Warning.GICv3_CommandInvalidEncoding

Fields:

ADDRESS unsigned int
: Address that the command was read from.

DATA unsigned int
: Data for the command.

ArchMsg.Warning.GICv3_CommandReadError

Fields:

ADDRESS unsigned int
: Address that the command was attepted to be read from.

ArchMsg.Warning.GICv3_DISCARDInvalidDevice

Fields:

DEVICE_ID unsigned int
: Device ID for interrupt source.

ID unsigned int
: Interrupt ID that should be discarded.

ArchMsg.Warning.GICv3_DISCARDInvalidID

Fields:

DEVICE_ID unsigned int
: Device ID for interrupt source.

ID unsigned int
: Interrupt ID that should be discarded.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ITT_ADDRESS unsigned int
The ITT address found in the device table.

ArchMsg.Warning.GICv3_GITS_BASER_AddressAlignementError

Fields:

BASER_INDEX unsigned int
Index of the BASER register that should have been provisioned.
BASE_ADDRESS unsigned int
The base address for the table
PAGE_SIZE_KB unsigned int
The size of a page in kB

ArchMsg.Warning.GICv3_GITS_BASER_CollectionTableLargerThanSupportedCollectionSpan

Fields:

BASER_INDEX unsigned int
Index of the BASER register that should have been provisioned.
BASER_TYPE enum
The type of the BASER register that needs to have been provisioned.
COLECTION_COUNT_IN_MEM unsigned int
The number of collections suported by this ITS
COLLECTION_ENTRY_SIZE unsigned int
The size in bytes of a collection entry
MAX_NEEDED_PAGE_COUNT unsigned int
The number of pages needed to hold all memory based supported collections
PROVISIONED_PAGE_COUNT unsigned int
The number of provisioned pages

ArchMsg.Warning.GICv3_GITS_BASER_InsufficientMemory

Fields:

BASER_INDEX unsigned int
Index of the BASER register that should have been provisioned.
BASER_TYPE enum
The type of the BASER register that needs to have been provisioned.
ID unsigned int
ID being used to store information.
NAME string
Command or register name that caused device to be used.

ArchMsg.Warning.GICv3_GITS_BASER_InvalidIndirectEntry

Fields:

BASER_INDEX unsigned int
Index of the BASER register that should have been provisioned.
BASER_TYPE enum
The type of the BASER register that needs to have been provisioned.
ID unsigned int
ID being used to store information.
NAME string
Command or register name that caused device to be used.

ArchMsg.Warning.GICv3_GITS_BASER_NotProvisioned

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

BASER_INDEX unsigned int
: Index of the BASER register that should have been provisioned.

BASER_TYPE enum
: The type of the BASER register that needs to have been provisioned.

NAME string
: Command or register name that caused device to be used.

ArchMsg.Warning.GICv3_GITS_CTLR_UnpredWrite

Fields:

ENABLE bool
: Value of the current GITS_CTLR.Enable

ITSNUM unsigned int
: Current value ot GITS_CTLR.ITSNumber

QUIESCENT bool
: Value of the current GITS_CTLR.Quiescent

VALUE unsigned int
: Value being written to the register, after masking

ArchMsg.Warning.GICv3_INVALLInvalid

Fields:

COLLECTION_ID unsigned int
: Collection ID to be invalidated.

ArchMsg.Warning.GICv3_INVInvalidCollection

Fields:

COLLECTION_ID unsigned int
: Collection ID that cannot be found.

DEVICE_ID unsigned int
: Device ID for interrupt source.

ID unsigned int
: Interrupt ID that should be invalidated.

ITT_ADDRESS unsigned int
: The ITT address found in the device table.

ArchMsg.Warning.GICv3_INVInvalidDevice

Fields:

DEVICE_ID unsigned int
: Device ID for interrupt source.

ID unsigned int
: Interrupt ID that should be inved.

ArchMsg.Warning.GICv3_INVInvalidID

Fields:

DEVICE_ID unsigned int
: Device ID for interrupt source.

ID unsigned int
: Interrupt ID that should be invalidated.

ITT_ADDRESS unsigned int
: The ITT address found in the device table.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.GICv3_ITS.MemoryMapped_ReadIgnored

Fields:

OFFSET unsigned int
: Offset of address within the GICv3 module

ArchMsg.Warning.GICv3_ITS.MemoryMapped_ReadReserved

Fields:

OFFSET unsigned int
: Offset of address within the GICv3 module

ArchMsg.Warning.GICv3_ITS.MemoryMapped_ReadWriteOnlyReg

Fields:

OFFSET unsigned int
: Offset of address within the GICv3 module

REG_NAME string
: Name of the register being read

ArchMsg.Warning.GICv3_ITS.MemoryMapped_WriteIgnored

Fields:

OFFSET unsigned int
: Offset of address within the GICv3 module

ArchMsg.Warning.GICv3_ITS.MemoryMapped_WriteReadOnlyReg

Fields:

OFFSET unsigned int
: Offset of address within the GICv3 module

REG_NAME string
: Name of the register being written

VALUE unsigned int
: Value written

ArchMsg.Warning.GICv3_ITS.MemoryMapped_WriteReserved

Fields:

OFFSET unsigned int
: Offset of address within the GICv3 module

VALUE unsigned int
: Value written

ArchMsg.Warning.GICv3_ITS_TranslateDeviceOutOfRangeIgnored

Fields:

DEVICE_ID unsigned int
: Device ID to be mapped.

DEVICE_ID_BITS unsigned int
: Device ID bits supported

ArchMsg.Warning.GICv3_ITS_TranslateEventOutOfRangeIgnored

Fields:

DEVICE_ID unsigned int
: ID of the device trigerring the event

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

EVENT_ID unsigned int
: Event ID to be mapped

EVENT_ID_BITS unsigned int
: Event ID bits supported

ArchMsg.Warning.GICv3_ITS_TranslateRequestIgnored

Fields:

DEVICE_ID unsigned int
: Device ID to be mapped.

EVENT_ID unsigned int
: Event ID to be mapped

SYNDROME enum
: Error Syndrome

ArchMsg.Warning.GICv3_InvalidDeviceMap

Fields:

CMD_NAME string
: The name of ITS Command that triggered the translation.

DEVICE_ID unsigned int
: Non-mapped device ID specified in the command.

ID unsigned int
: Incoming interrupt ID specified in the command.

ArchMsg.Warning.GICv3_InvalidTargetAddress

Fields:

COMMAND_NAME string
: Name of the command being executed.

PROCESSOR_NUM unsigned int
: Processor number specified in the command.

ArchMsg.Warning.GICv3_InvalidVCPU

Fields:

CMD_NAME string
: The name of ITS Command that requested the map.

VCPU unsigned int
: Virtual CPU specified in the command.

ArchMsg.Warning.GICv3_LPIUnmappedCollection

Fields:

COLLECTION_ID unsigned int
: Collection ID that the interrupt maps to.

DEVICE_ID unsigned int
: Device ID for the interrupt.

ID unsigned int
: Incoming interrupt ID specified.

ITT_ADDRESS unsigned int
: The ITT address found in the device table.

SET bool
: Whether the interrupt is set (rather than cleared).

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.GICv3_LPIUnmappedDevice

Fields:

DEVICE_ID unsigned int
: Device ID for the interrupt.

ID unsigned int
: Incoming interrupt ID specified.

SET bool
: Whether the interrupt is set (rather than cleared).

ArchMsg.Warning.GICv3_LPIUnmappedID

Fields:

DEVICE_ID unsigned int
: Device ID for the interrupt.

ID unsigned int
: Incoming interrupt ID specified.

ITT_ADDRESS unsigned int
: The ITT address found in the device table.

SET bool
: Whether the interrupt is set (rather than cleared).

ArchMsg.Warning.GICv3_MAPCUnmapInvalid

Fields:

COLLECTION_ID unsigned int
: Collection ID requested to be unmapped.

ArchMsg.Warning.GICv3_MAPCUnmapInvalidTarget

MAPC command request unmapping with non-zero target value.

ArchMsg.Warning.GICv3_MAPDUnmapInvalid

Fields:

DEVICE_ID unsigned int
: Device ID attempting to be unmapped.

ArchMsg.Warning.GICv3_MAPDUnmapInvalidSize

MAPD command request unmapping with non-zero Size value.

ArchMsg.Warning.GICv3_MAPIInvalid

Fields:

COLLECTION_ID unsigned int
: Collection ID specified in the map request.

DEVICE_ID unsigned int
: Non-mapped device ID specified in the map request.

ID unsigned int
: Incoming interrupt ID specified in the map request.

PHYSICAL_ID unsigned int
: Outgoing interrupt ID specified in the map request.

ArchMsg.Warning.GICv3_MOVIInvalidCollection

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DEVICE_ID unsigned int
: Device ID for interrupt source.

ID unsigned int
: Input interrupt ID that should be moved.

ITT_ADDRESS unsigned int
: The ITT address found in the device table.

NEW_COLLECTION_ID unsigned int
: Collection ID that was requested.

NEW_INVALID bool
: It is the new collection ID that is invalid (rather than the old collection ID).

OLD_COLLECTION_ID unsigned int
: Collection ID that this interrupt used to be associated with.

ArchMsg.Warning.GICv3_MOVIInvalidDevice

Fields:

DEVICE_ID unsigned int
: Device ID for interrupt source.

ID unsigned int
: Input interrupt ID that should be moved.

NEW_COLLECTION_ID unsigned int
: Collection ID that was requested.

ArchMsg.Warning.GICv3_MOVIInvalidID

Fields:

DEVICE_ID unsigned int
: Device ID for interrupt source.

ID unsigned int
: Input interrupt ID that should be moved.

ITT_ADDRESS unsigned int
: The ITT address found in the device table.

NEW_COLLECTION_ID unsigned int
: Collection ID that was requested.

ArchMsg.Warning.GICv3_RangeErrorCollection

Fields:

COLLECTION_ID unsigned int
: Collection ID specified.

NAME string
: Command or register name.

ArchMsg.Warning.GICv3_RangeErrorDeviceID

Fields:

DEVICE_ID unsigned int
: Device ID specified.

NAME string
: Command or register name.

ArchMsg.Warning.GICv3_RangeErrorIDBits

Fields:

ID_BITS unsigned int
: Number of ID bits specified.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NAME string
: Command or register name.

ArchMsg.Warning.GICv3_RangeErrorIncomingID

Fields:

COLLECTION_ID unsigned int
: Collection ID specified.

ID_BITS unsigned int
: Maximum number of ID bits allowed for this access.

NAME string
: Command or register name.

ArchMsg.Warning.GICv3_WriteIgnoredConfiguration

Fields:

REG_NAME string
: Name of register being read or written.

VALUE unsigned int
: Value attempting to be written.

CHECKPOINT_MESSAGE

Report error messages from the checkpointing process. Fields:

message string
: Message contents.

CHECKPOINT_RESTORE_END

Checkpoint restore completed.

CHECKPOINT_RESTORE_START

Checkpoint restore about to start.

CHECKPOINT_SAVE_END

Checkpoint save completed.

CHECKPOINT_SAVE_START

Checkpoint save about to start.

GICv3_CFGSDISABLE

Fields:

STATE bool
: The new state of the signal.

GICv3_CommandComplete

Fields:

COMMAND_TYPE enum
: Type of the command being executed.

ERROR enum
: The error type if the command was not successful.

SUCCESS bool
: Whether the command succeeded.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

GICv3_CommandDecode

Fields:

COMMAND_NAME string
: Name of the decoded command.

DATA_0 unsigned int
: First 64-bit word from the command.

DATA_1 unsigned int
: Second 64-bit word from the command.

DATA_2 unsigned int
: Third 64-bit word from the command.

DATA_3 unsigned int
: Forth 64-bit word from the command.

OFFSET unsigned int
: Offset from GITS_CBASER that the command was read from.

VALID bool
: Whether the command was valid.

GICv3_CommandStart_CLEAR

Fields:

DEVICE unsigned int
: The Device ID specified in the command.

ID unsigned int
: The incoming ID of interrupts to move to a new collection.

GICv3_CommandStart_DISCARD

Fields:

DEVICE unsigned int
: The Device ID specified in the command.

ID unsigned int
: The incoming ID of interrupts to discard.

GICv3_CommandStart_INT

Fields:

DEVICE unsigned int
: The Device ID specified in the command.

ID unsigned int
: The incoming ID of interrupts to move to a new collection.

GICv3_CommandStart_INV

Fields:

DEVICE unsigned int
: The Device ID specified in the command.

ID unsigned int
: The incoming ID of interrupts to invalidate caches for.

GICv3_CommandStart_INVALL

Fields:

COLLECTION unsigned int
: The collection for which all interrupt caches should be invalidated.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

GICv3_CommandStart_MAPC

Fields:

COLLECTION unsigned int
: The ID of the collection being mapped.

TARGET unsigned int
: The processor number or redistributor base address for the redistributor to synchronise.

VALID bool
: Whether to create a new device mapping (true) or to discard an old one (false).

GICv3_CommandStart_MAPD

Fields:

DEVICE unsigned int
: The Device ID specified in the command.

ITT_ADDRESS unsigned int
: The interrupt translation table address specified in the command.

SIZE unsigned int
: Number of interrupt ID bits supported by this device, minus one.

VALID bool
: Whether to create a new device mapping (true) or to discard an old one (false).

GICv3_CommandStart_MAPI

Fields:

COLLECTION unsigned int
: The ID of the collection to map the interrupts to.

DEVICE unsigned int
: The Device ID specified in the command.

ID unsigned int
: The incoming ID of interrupts to map (and the same ID that will be passed to software).

GICv3_CommandStart_MAPVI

Fields:

COLLECTION unsigned int
: The ID of the collection to map the interrupts to.

DEVICE unsigned int
: The Device ID specified in the command.

ID unsigned int
: The incoming ID of interrupts to map.

PHYSICAL_ID unsigned int
: The ID presented to software for these interrupts.

GICv3_CommandStart_MOVALL

Fields:

SOURCE unsigned int
: The processor number or redistributor base address for the source redistributor.

TARGET unsigned int
: The processor number or redistributor base address for the target redistributor.

GICv3_CommandStart_MOVI

Fields:

COLLECTION unsigned int
: The ID of the new collection to move the interrupt to.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DEVICE unsigned int
: The Device ID specified in the command.

ID unsigned int
: The incoming ID of interrupts to move to a new collection.

GICv3_CommandStart_SYNC

Fields:

TARGET unsigned int
: The processor number or redistributor base address for the redistributor to synchronise.

GICv3_CommandStart_VINVALL

Fields:

VCPU unsigned int
: The VCPU id for which all interrupt caches should be invalidated.

GICv3_CommandStart_VMAPI

Fields:

DEVICE unsigned int
: The Device ID specified in the command.

ID unsigned int
: The incoming ID of interrupt to map.

PHYSICAL_ID unsigned int
: The doorbell ID to be presented to guest

VCPU unsigned int
: The VCPU ID specified in the command.

GICv3_CommandStart_VMAPP

Fields:

SIZE unsigned int
: Number of interrupt ID bits supported by this device, minus one.

TARGET_ADDRESS unsigned int
: The Redistributor address specified in the command.

VALID bool
: Whether to create a new device mapping (true) or to discard an old one (false).

VCPU unsigned int
: The VCPU ID specified in the command.

VPT_ADDRESS unsigned int
: The Virtual Pending Table address specified in the command.

GICv3_CommandStart_VMAPVI

Fields:

DEVICE unsigned int
: The Device ID specified in the command.

ID unsigned int
: The incoming ID of interrupt to map.

PHYSICAL_ID unsigned int
: The doorbell ID to be presented to guest

VCPU unsigned int
: The VCPU ID specified in the command.

VIRTUAL_ID unsigned int
: The Virtual ID to be presented to guest.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

GICv3_CommandStart_VMOVI

Fields:

DEVICE unsigned int
: The Device ID specified in the command.

ID unsigned int
: The incoming ID of interrupt to map.

PHYSICAL_ID unsigned int
: The doorbell ID to be presented to guest

VCPU unsigned int
: The VCPU ID specified in the command.

GICv3_CommandStart_VMOVP

Fields:

ITS_LIST unsigned int
: The ITS numbers participating in the synchronizatio operation.

SEQUENCE_NUM unsigned int
: The identifier of synchronizatio point in the command.

TARGET_ADDRESS unsigned int
: The Redistributor address specified in the command.

VCPU unsigned int
: The VCPU ID specified in the command.

GICv3_CommandStart_VSYNC

Fields:

VCPU unsigned int
: The VCPU id for which commands must be synchronized.

GICv3_DISCARD

Fields:

DEVICE_ID unsigned int
: Device ID for interrupt source.

ID unsigned int
: Interrupt ID that should be discarded.

ITT_ADDRESS unsigned int
: The ITT address found in the device table.

GICv3_INV

Fields:

COLLECTION_ID unsigned int
: Collection ID for this interrupt.

DEVICE_ID unsigned int
: Device ID for interrupt source.

ID unsigned int
: Input interrupt ID that should be invalidated.

ITT_ADDRESS unsigned int
: The ITT address found in the device table.

PHYSICAL_ID unsigned int
: Interrupt ID that will be specified to the redistributor.

REDISTRIBUTOR_BASE unsigned int
: Base address for the redistributor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

GICv3_INVALL

Fields:

COLLECTION_ID unsigned int
: Collection ID to be invalidated.

REDISTRIBUTOR_BASE unsigned int
: Base address for the redistributor.

GICv3_ITS.MemoryMapped_Read

Trace read from a GICv3 memory-mapped register. Fields:

A0 unsigned int
: Affinity level 0 (for banked registers)

A1 unsigned int
: Affinity level 1 (for banked registers)

A2 unsigned int
: Affinity level 2 (for banked registers)

A3 unsigned int
: Affinity level 3 (for banked registers)

BANKED bool
: Whether the access is to a banked register

NS bool
: Access is non-secure

OFFSET unsigned int
: Offset of address within the GICv3 module

REG_NAME string
: Name of the register being read

VALUE unsigned int
: Value returned

GICv3_ITS.MemoryMapped_Read64

Trace read from a GICv3 memory-mapped register. Fields:

A0 unsigned int
: Affinity level 0 (for banked registers)

A1 unsigned int
: Affinity level 1 (for banked registers)

A2 unsigned int
: Affinity level 2 (for banked registers)

A3 unsigned int
: Affinity level 3 (for banked registers)

BANKED bool
: Whether the access is to a banked register

NS bool
: Access is non-secure

OFFSET unsigned int
: Offset of address within the GICv3 module

REG_NAME string
: Name of the register being read

VALUE unsigned int
: Value returned

GICv3_ITS.MemoryMapped_Write

Trace write to a GICv3 memory-mapped register. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

A0 unsigned int
: Affinity level 0 (for banked registers)

A1 unsigned int
: Affinity level 1 (for banked registers)

A2 unsigned int
: Affinity level 2 (for banked registers)

A3 unsigned int
: Affinity level 3 (for banked registers)

BANKED bool
: Whether the access is to a banked register

NS bool
: Access is non-secure

OFFSET unsigned int
: Offset of address within the GICv3 module

REG_NAME string
: Name of the register being written

UPDATED_VALUE unsigned int
: New value read back from the register; for write-only registers this will be equal to VALUE.

VALUE unsigned int
: Value written

GICv3_ITS.MemoryMapped_Write64

Trace write to a GICv3 memory-mapped register. Fields:

A0 unsigned int
: Affinity level 0 (for banked registers)

A1 unsigned int
: Affinity level 1 (for banked registers)

A2 unsigned int
: Affinity level 2 (for banked registers)

A3 unsigned int
: Affinity level 3 (for banked registers)

BANKED bool
: Whether the access is to a banked register

NS bool
: Access is non-secure

OFFSET unsigned int
: Offset of address within the GICv3 module

REG_NAME string
: Name of the register being written

UPDATED_VALUE unsigned int
: New value read back from the register; for write-only registers this will be equal to VALUE.

VALUE unsigned int
: Value written

GICv3_ITS.MemoryMapped_WritePort

Trace write to a GICv3 memory-mapped register acting as imp def port. Fields:

FIRST_BYTE_IN_BURST unsigned int
: First byte written in the burst

NS bool
: Access is non-secure

OFFSET unsigned int
: Offset of address within the GICv3 module

REG_NAME string
: Name of the register being written

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

GICv3_ITS_TranslationStart

Fields:

DEVICE_ID unsigned int
: Device ID to be mapped.

EVENT_ID unsigned int
: Event ID to be mapped

START bool
: Translation Started

GICv3_ITS_TranslationStatus

Fields:

DEVICE_ID unsigned int
: Device ID to be mapped.

EVENT_ID unsigned int
: Event ID to be mapped

SUCCESS bool
: Translation Success

GICv3_LPISent

Fields:

COLLECTION_ID unsigned int
: Collection ID that the interrupt maps to.

DEVICE_ID unsigned int
: Device ID for the interrupt.

ID unsigned int
: Incoming interrupt ID specified.

ITT_ADDRESS unsigned int
: The ITT address found in the device table.

PHYSICAL_ID unsigned int
: Outgoing interrupt ID found in the map.

REDISTRIBUTOR_BASE unsigned int
: Base address for redistristributor (may be an internal address for monolithic implementations).

SET bool
: Whether the interrupt is set (rather than cleared).

GICv3_MAPCMap

Fields:

ADDRESS unsigned int
: Redistributor base address.

COLLECTION_ID unsigned int
: Collection ID to be mapped.

GICv3_MAPCMapMonolithic

Fields:

ADDRESS unsigned int
: Redistributor base address.

COLLECTION_ID unsigned int
: Collection ID to be mapped.

PROC_NUM unsigned int
: Processor number for redistributor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

GICv3_MAPCUnmap

Fields:

COLLECTION_ID unsigned int
: Collection ID being unmapped.

GICv3_MAPDMap

Fields:

DEVICE_ID unsigned int
: Device ID to be mapped.

INTERRUPT_BITS unsigned int
: Number of interrupt ID bits supported by the device.

ITT_ADDRESS unsigned int
: Base address for memory to store Interrupt Translation Table.

GICv3_MAPDUnmap

Fields:

DEVICE_ID unsigned int
: Device ID to be unmapped.

GICv3_MAPI

Fields:

COLLECTION_ID unsigned int
: Collection ID specified in the map request.

DEVICE_ID unsigned int
: Device ID specified in the map request.

ID unsigned int
: Incoming interrupt ID specified in the map request.

ITT_ADDRESS unsigned int
: The ITT address found in the device table.

PHYSICAL_ID unsigned int
: Outgoing interrupt ID specified in the map request.

GICv3_MOVALL

Fields:

NEW_REDISTRIBUTOR_BASE unsigned int
: Base address (or processor number for monolithic implementation) for destination redistristributor.

OLD_REDISTRIBUTOR_BASE unsigned int
: Base address (or processor number for monolithic implementation) for source redistristributor.

GICv3_MOVI

Fields:

DEVICE_ID unsigned int
: Device ID for interrupt source.

ID unsigned int
: Input interrupt ID that was moved.

ITT_ADDRESS unsigned int
: The ITT address found in the device table.

MOVE_REQUIRED bool
: Whether a move between redistributors was required.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NEW_COLLECTION_ID unsigned int
: Collection ID that this interrupt is now associated with.

OLD_COLLECTION_ID unsigned int
: Collection ID that this interrupt used to be associated with.

PHYSICAL_ID unsigned int
: Interrupt ID that will be specified to the redistributor.

GICv3_MultiChip_Forwarding_LPI_CMD_ACK_to_Dist

Fields:

THIS_CHIP_ID unsigned int
: Chip

GICv3_MultiChip_Forwarding_LPI_CMD_REQ_to_Dist

Fields:

CMD_TYPE unsigned int
: The command type

DST_CHIP unsigned int
: Destination chip for the command

DST_CORE unsigned int
: Destination core for the command

LPI_ID unsigned int
: LPI interrupt ID being operated on by the command

SRC_CHIP unsigned int
: Source chip for the command

SRC_CORE unsigned int
: Source core for the command

THIS_CHIP_ID unsigned int
: Chip ID

GICv3_MultiChip_SETLPI_BY_MOVI

Fields:

DST_TGT unsigned int
: Destination chip and core for MOVI

LPI unsigned int
: LPI to be moved

SRC_TGT unsigned int
: Source chip and core for MOVI

THIS_CHIP_ID unsigned int
: Chip

GICv3_MultiChip_VMOVP_Received

Fields:

DST_TGT unsigned int
: Destination chip and core for VMOVP

SRC_TGT unsigned int
: Source chip and core for VMOVP

THIS_CHIP_ID unsigned int
: Chip

vPEID unsigned int
: ID of the vPE to be moved

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

GICv3_RegUpdated32

Fields:

REG_NAME string
: The name of the register being updated.

VALUE unsigned int
: The updated value.

GICv3_RegUpdated64

Fields:

REG_NAME string
: The name of the register being updated.

VALUE unsigned int
: The updated value.

GICv3_TableWalk_Collection

Fields:

ID unsigned int
: The ID of the collection being looked up.

TARGET unsigned int
: The target address (or processor number) specified for this collection.

GICv3_TableWalk_CollectionInvalid

Fields:

ID unsigned int
: The ID of the collection being looked up.

GICv3_TableWalk_Device

Fields:

DEVICE_ID unsigned int
: The device ID being looked up.

INTERRUPT_BITS unsigned int
: The number of interrupt bits specified in the device table.

ITT_ADDRESS unsigned int
: The ITT address found in the device table.

GICv3_TableWalk_DeviceInvalid

Fields:

DEVICE_ID unsigned int
: The device ID being looked up.

GICv3_TableWalk_ITT

Fields:

COLLECTION_ID unsigned int
: The collection ID used for routing this interrupt to the appropriate CPU.

ID unsigned int
: The input indentifier used for the translation process.

ITT_ADDRESS unsigned int
: The base address specified in the previous MAPD command for this table.

PHYSICAL_ID unsigned int
: The physical interrupt ID presented to software.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

GICv3_TableWalk_ITTInvalid

Fields:

ID unsigned int
: The input indentifier used for the translation process.

ITT_ADDRESS unsigned int
: The base address specified in the MAPD command that is being used.

GICv3_TableWalk_ITT_VLPI

Fields:

ID unsigned int
: The input indentifier used for the translation process.

ITT_ADDRESS unsigned int
: The base address specified in the previous MAPD command for this table.

PHYSICAL_ID unsigned int
: The physical interrupt ID presented to software.

VCPU unsigned int
: The VCPU ID used for routing this interrupt to the appropriate CPU.

GICv3_TableWalk_VCPU

Fields:

TARGET unsigned int
: The Redistributor address specified in VCPU table.

VCPU unsigned int
: The VCPU being looked up.

VINTERRUPT_BITS unsigned int
: The number of virtual interrupt bits specified in the VCPU table.

VPT_ADDRESS unsigned int
: The Virtual Pending Table address assigned to VCPU.

GICv3_TableWalk_VCPUInvalid

Fields:

VCPU unsigned int
: The Virtual CPU ID being looked up.

GICv3_TableWalk_VPTAddrInvalid

Fields:

VPTAddr unsigned int
: The Virtual Pending Table address supposedly corresponding to a VCPU, being looked up.

GICv3_VMAPI

Fields:

DEVICE_ID unsigned int
: Device ID specified in the map request.

ID unsigned int
: Incoming interrupt ID specified in the map request.

ITT_ADDRESS unsigned int
: The ITT address found in the device table.

PHYSICAL_ID unsigned int
: Outgoing Physical interrupt ID specified in the map request.

VCPU unsigned int
: Virtual CPU specified in the map request.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VIRTUAL_ID unsigned int
: Outgoing Virtual interrupt ID specified in the map request.

GICv3_VMAPP_Map

Fields:

TARGET_ADDRESS unsigned int
: The Redistributor address assigned to VCPU.

VCPU unsigned int
: VCPU ID to be mapped.

VPT_ADDRESS unsigned int
: The Virtual Pending Table address assigned to VCPU.

GICv3_VMAPP_Unmap

Fields:

VCPU unsigned int
: Virtual CPU being unmapped.

GICv3_VMOVI

Fields:

DEVICE_ID unsigned int
: Device ID for interrupt source.

ID unsigned int
: Input interrupt ID that was moved.

MOVE_REQUIRED bool
: Whether a move between redistributors was required.

NEW_VCPU unsigned int
: Collection ID that this interrupt is now associated with.

OLD_VCPU unsigned int
: Collection ID that this interrupt used to be associated with.

PHYSICAL_ID unsigned int
: DoorBell Interrupt ID that will be specified to the redistributor.

VIRTUAL_ID unsigned int
: Interrupt ID that will be specified to the redistributor.

GICv4_ITS_VMOVPSynchronization

Fields:

ITSNUM unsigned int
: Current value of GITS_CTLR.ITSNumber.

REDISTRIBUTOR_BASE unsigned int
: The base address of the new redistributor that the VCPU should be mapped to.

SENDINGnRECEIVED bool
: Whether the ITS is sending or has received a VMOVP to/from the router unit.

VCPU unsigned int
: The VCPU to be remapped.

GICv4_ITS_VMOVPSynchronizationAck

Fields:

ITSNUM unsigned int
: Current value of GITS_CTLR.ITSNumber.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

GICv4_ITS_VMOVPSynchronizationDone

Fields:

ITSNUM unsigned int
: Current value of GITS_CTLR.ITSNumber.

GICv4_ITS_VMOVPSynchronizationStarted

Fields:

ITSNUM unsigned int
: Current value of GITS_CTLR.ITSNumber.

RDBase unsigned int
: The base address of the new redistributor that the VCPU should be mapped to.

VCPU unsigned int
: The VCPU to be remapped.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.79 GICv3ProtocolChecker - trace

This section describes the trace sources.

GICv3ProtocolChecker.protocol_violation

Fields:

MESSAGE `string`
Protocol Violation Message

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.80 GICv3Redistributor - trace

This section describes the trace sources.

ArchMsg.Error.GICv3IgnorePPILegacyPendingNoSource

Fields:

INTERRUPT_ID unsigned int
: The interrupt ID of the PPI

ArchMsg.Error.GICv3_UnexpectedClearAcknowledge

Received a ClearAcknowlege command but not expecting one.

ArchMsg.Error.GICv3_UnknownTypeUpstreamWrite

Upstream write of unknown type received. Acknowledging and ignoring.This is a PROTOCOL ERROR, the architecture allows system behaviour to become UNPREDICATBLE from this point onwards.

ArchMsg.Error.LPIConfigReadError

Fields:

ADDRESS unsigned int
: Address used for the access

INTERRUPT_ID unsigned int
: The interrupt ID for the LPI

ArchMsg.Error.LPIDeviceWriteError

Fields:

ADDRESS unsigned int
: Address used for the access

ArchMsg.Error.LPIPendingAccessError

Fields:

ADDRESS unsigned int
: Address used for the access

READnWRITE bool
: Whether this is a read or a write

ArchMsg.Info.GICv3_DroppedInternalPacket

Fields:

INTERFACE unsigned int
: Index of the interface

ArchMsg.Info.GICv3_LPI_CachedProperty_Discard

Fields:

INTERRUPT_ID unsigned int
: The interrupt ID for the LPI

ArchMsg.Info.GICv3_LPI_CachedProperty_Update

Fields:

INTERRUPT_ID unsigned int
: The interrupt ID for the LPI

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

PRIORITY unsigned int
The priority for the Cached interrupt (unshifted value)

ArchMsg.Warning.GICv3-PITS.DeactivateIgnored

Fields:

INTERRUPT_ID unsigned int
The interrupt ID in the EOI message

ArchMsg.Warning.GICv3-PITS.TrackActive-NotFound

Fields:

INTERRUPT_ID unsigned int
The interrupt ID in the Activate message

ArchMsg.Warning.GICv3.GICR_CTLR_EnableLPI_negedge_ignored

Update to GICR_CTLR has EnableLPI as 0, but EnableLPI has been set to 1 in the past, and this bit only goes one way.

ArchMsg.Warning.GICv3.GenerateSGI_A3VnotSupported

GenerateSGI packet received form CPU with A3V set, the Redistributor does not support A3, it will be acknowledged but is ignored.

ArchMsg.Warning.GICv3.GenerateSGI_IRMwithA3VorAff

GenerateSGI packet received form CPU with IRM=1 but A3V or one of the Ax fields is not zero, this packet should not exist but will be handled as if A0=A1=A2=A3=0.

ArchMsg.Warning.GICv3.GenerateSGI_Ignored

GenerateSGI packet received form CPU but ARE_S and ARE_NS are 0 and distributor is configured to ignore such packets.

ArchMsg.Warning.GICv3.PENDBASERUpdateIgnored

Update to PENDBASER when GICR_CTLR.EnableLPI=1 and GICR_WAKER.Quiescent=0 is ignored.

ArchMsg.Warning.GICv3.PROPBASERUpdateIgnored

Update to PROPBASER when GICR_CTLR.EnableLPI=1 and GICR_WAKER.Quiescent=0 is ignored.

ArchMsg.Warning.GICv3.PROPBASERUpdateReadOnly

Update to PROPBASER when register is read only -- Has been ignored and probably won't perform as you would expect.

ArchMsg.Warning.GICv3IgnorePPINotImplemented

Fields:

INTERRUPT_ID unsigned int
The interrupt ID of the PPI

ArchMsg.Warning.GICv3_A3NonZeroNotSupported

Fields:

A3 unsigned int
Affinity level 3 value.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.GICv3_CFGSDISABLE_unsupported

CFGSDISABLE signal has been set but doing so has no effect.

ArchMsg.Warning.GICv3_DroppedInternalPacket

Fields:

INTERFACE unsigned int
: Index of the interface

ArchMsg.Warning.GICv3_GICR_WAKER.PROCESSOR_SLEEP clear ignored

GICR_WAKER.PROCESSOR_SLEEP clear ignored as SLEEP is true.

ArchMsg.Warning.GICv3_GICR_WAKER.SLEEP set ignored

GICR_WAKER.SLEEP set ignored as PROCESSOR_SLEEP is not true.

ArchMsg.Warning.GICv3_IgnoredActivate

Received Activate packet with unusable payload (8196>ID>1023 and 9:4 != 0).

ArchMsg.Warning.GICv3_IgnoredActivateARE

Received Activate packet with unusual payload (8196>ID>1023) and 9:4 == 0 but ARE forbids the packet to be considered.

ArchMsg.Warning.GICv3_IgnoredDeactivate

Received Deactivate packet with unusable payload (8196>ID>1023 and 9:4 != 0).

ArchMsg.Warning.GICv3_IgnoredDeactivateARE

Received Deactivate packet with unusual payload (8196>ID>1023) and 9:4 == 0 but ARE forbids the packet to be considered.

ArchMsg.Warning.GICv3_IgnoredDeactivateSpurious

Received Deactivate packet with a special ID (1020-1023) or an out of range ID.

ArchMsg.Warning.GICv3_LPIDroppedOutOfRange

Fields:

INTERRUPT_ID unsigned int
: The interrupt ID for the LPI

REG_NAME string
: Name of the register being written

ArchMsg.Warning.GICv3_LPI_CachedProperty_CacheMismatch

Fields:

CACHED_CFG unsigned int
: Configuration the cache returned

INTERRUPT_ID unsigned int
: The interrupt ID for the LPI

MEMORY_CFG unsigned int
: Configuration present in memory for the LPI

ArchMsg.Warning.GICv3_LPIsDisabled

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

READnWRITE bool
: Whether the access was a read.

REG_NAME string
: Name of the register being accessed.

ArchMsg.Warning.GICv3_MemoryMapped_AccessInIncorrectAREState

Fields:

ARE_NS bool
: Value of GICD_CTLR.ARE_NS.

ARE_S bool
: Value of GICD_CTLR.ARE_S.

NS bool
: Access is non-secure.

READnWRITE bool
: Access is a read (rather than a write).

REG_NAME string
: Name of the register being accessed

ArchMsg.Warning.GICv3_MemoryMapped_IgnoredPendingChange

Fields:

INTERRUPT_ID unsigned int
: The interrupt ID specified in the register access.

IS_PENDING bool
: The current pending state of the interrupt (and what the access attempted to set it to).

REG_NAME string
: Name of the register being accessed.

ArchMsg.Warning.GICv3_MemoryMapped_NSAccessToSecureInterruptIgnored

Fields:

INTERRUPT_ID unsigned int
: The interrupt ID specified in the register access.

REG_NAME string
: Name of the register being accessed.

ArchMsg.Warning.GICv3_MemoryMapped_WriteIgnored

Fields:

REG_FIELD_NAME string
: Name of register and optionally field being written.

VALUE unsigned int
: Value attempting to be written.

ArchMsg.Warning.GICv3_ReceivedInvalidCommandFromAbove

Fields:

ACTUAL_LENGTH unsigned int
: Actual length of the data in bytes including the header.

DATA unsigned int
: The data received (including header).

EXPECTED_LENGTH unsigned int
: The length in bytes including header that was expected to be received dependent on command type and potentially the data type in the case of a data write.

TYPE unsigned int
: The command type.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.GICv3_ReceivedInvalidCommandFromBelow

Fields:

ACTUAL_LENGTH unsigned int
: Actual length of the data in bytes including the header.

DATA unsigned int
: The data received (including header).

EXPECTED_LENGTH unsigned int
: The length in bytes including header that was expected to be received dependent on command type and potentially the data type in the case of a data write.

REDISTRIBUTOR_INDEX unsigned int
: Index for the downstream port that this command was received on.

TYPE unsigned int
: The command type.

ArchMsg.Warning.GICv3_Redistributor.MemoryMapped_ReadIgnored

Fields:

OFFSET unsigned int
: Offset of address within the GICv3 module

ArchMsg.Warning.GICv3_Redistributor.MemoryMapped_ReadReserved

Fields:

OFFSET unsigned int
: Offset of address within the GICv3 module

ArchMsg.Warning.GICv3_Redistributor.MemoryMapped_ReadWriteOnlyReg

Fields:

OFFSET unsigned int
: Offset of address within the GICv3 module

REG_NAME string
: Name of the register being read

ArchMsg.Warning.GICv3_Redistributor.MemoryMapped_WriteIgnored

Fields:

OFFSET unsigned int
: Offset of address within the GICv3 module

ArchMsg.Warning.GICv3_Redistributor.MemoryMapped_WriteReadOnlyReg

Fields:

OFFSET unsigned int
: Offset of address within the GICv3 module

REG_NAME string
: Name of the register being written

VALUE unsigned int
: Value written

ArchMsg.Warning.GICv3_Redistributor.MemoryMapped_WriteReserved

Fields:

OFFSET unsigned int
: Offset of address within the GICv3 module

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VALUE unsigned int
Value written

ArchMsg.Warning.GICv3_Redistributor_Internal.MemoryMapped_ReadIgnored

Fields:

OFFSET unsigned int
Offset of address within the GICv3 module

ArchMsg.Warning.GICv3_Redistributor_Internal.MemoryMapped_ReadReserved

Fields:

OFFSET unsigned int
Offset of address within the GICv3 module

ArchMsg.Warning.GICv3_Redistributor_Internal.MemoryMapped_ReadWriteOnlyReg

Fields:

OFFSET unsigned int
Offset of address within the GICv3 module
REG_NAME string
Name of the register being read

ArchMsg.Warning.GICv3_Redistributor_Internal.MemoryMapped_WriteIgnored

Fields:

OFFSET unsigned int
Offset of address within the GICv3 module

ArchMsg.Warning.GICv3_Redistributor_Internal.MemoryMapped_WriteReadOnlyReg

Fields:

OFFSET unsigned int
Offset of address within the GICv3 module
REG_NAME string
Name of the register being written
VALUE unsigned int
Value written

ArchMsg.Warning.GICv3_Redistributor_Internal.MemoryMapped_WriteReserved

Fields:

OFFSET unsigned int
Offset of address within the GICv3 module
VALUE unsigned int
Value written

ArchMsg.Warning.GICv3_ReleasedNotSent

Fields:

INTERRUPT_ID unsigned int
The interrupt ID of the interrupt considered

ArchMsg.Warning.GICv3_SGIDiscardedDueToInvalidTarget

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

A1 unsigned int
: Affinity level 1.

A2 unsigned int
: Affinity level 2.

A3 unsigned int
: Affinity level 3.

INTERRUPT_ID unsigned int
: The ID of the SGI.

RANGE_SELECTOR unsigned int
: Value of the RS field used for generating the SGI if applicable, SBZ otherwise

TARGET_LIST unsigned int
: Bit vector of targets.

ArchMsg.Warning.GICv3_SGIRSNotSupported

IRI received Generate SGI packet from CPU with RSV=1 and RS non-zeroand the IRI does not support this feature. RS treated as 0.

ArchMsg.Warning.GICv3_SGI_Dropped

Fields:

ENABLED bool
: Whether the SGI is enable in the destination CPU.

INTERRUPT_G1 bool
: Whether the SGI is configured as group 1 in this redistributor.

INTERRUPT_ID unsigned int
: The interrupt ID of the SPI

INTERRUPT_NS bool
: Whether the SGI is configured as non-secure in this redistributor.

LEGACY bool
: IS the SGI a legacy type interrupt

NS bool
: Original register access was non-secure.

NSACR unsigned int
: The value of the NSACR field for the SGI

SGT enum
: The SGT field in the incoming GenerateSGI packet (corresponds to the ICC register that was written).

ArchMsg.Warning.GICv3_SGI_GICD_SGIR_BRoadcast_Dropped

Fields:

ENABLED bool
: Whether the SGI is enable in the destination CPU.

INTERRUPT_G1 bool
: Whether the SGI is configured as group 1 in this redistributor.

INTERRUPT_ID unsigned int
: The interrupt ID of the SPI

INTERRUPT_NS bool
: Whether the SGI is configured as non-secure in this redistributor.

NS bool
: Original register access was non-secure.

NSACR unsigned int
: The value of the NSACR field for the SGI

ArchMsg.Warning.GICv3_SPIDropped

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

A0 unsigned int
: Affinity level 0 address

A1 unsigned int
: Affinity level 1 address

A2 unsigned int
: Affinity level 2 address

A3 unsigned int
: Affinity level 3 address

INTERRUPT_ID unsigned int
: The interrupt ID for the LPI

ArchMsg.Warning.GICv3_VLPIDroppedOutOfRange

Fields:

INTERRUPT_ID unsigned int
: The interrupt ID for the LPI

REG_NAME string
: Name of the register being written

VPT unsigned int
: The address of the virtual pending table

ArchMsg.Warning.GICv3_VLPIsDisabled

Fields:

READnWRITE bool
: Whether the access was a read.

REG_NAME string
: Name of the register being accessed.

ArchMsg.Warning.GICv3_WakerequestIgnored

Wakerequest was asserted but the component is not asleep.

ArchMsg.Warning.GICv3_access_asleep_rd

Fields:

REG_NAME string
: Name of the register being accessed.

ArchMsg.Warning.GICv3_command_received_processorsleep

Command received from CPU when ProcessorSleep and ChildrenAsleep.

GICv3-PITS.TrackActive

Fields:

INTERRUPT_ID unsigned int
: The interrupt ID of the interrupt considered

PRIORITY unsigned int
: The interrupt Priority of the interrupt considered

GICv3-PITS.TrackPending

Fields:

INTERRUPT_ID unsigned int
: The interrupt ID of the interrupt considered

PRIORITY unsigned int
: The interrupt Priority of the interrupt considered

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

GICv3-PITS.TrackSent

Fields:

INTERRUPT_ID unsigned int
: The interrupt ID of the interrupt considered

PRIORITY unsigned int
: The interrupt Priority of the interrupt considered

GICv3_CFGSDISABLE

Fields:

STATE bool
: The new state of the signal.

GICv3_ClearDropped

Fields:

A0 unsigned int
: Affinity level 0 address

A1 unsigned int
: Affinity level 1 address

A2 unsigned int
: Affinity level 2 address

A3 unsigned int
: Affinity level 3 address

INTERRUPT_ID unsigned int
: The interrupt ID for the LPI

GICv3_InterruptSetPendingButGroupNotEnabled

Fields:

GROUP1 bool
: Whether the interrupt is group 1.

INTERRUPT_ID unsigned int
: The interrupt ID that was set pending.

NS bool
: Whether the interrupt is non-secure.

GICv3_InterruptSetReservedGroup

Fields:

INTERRUPT_ID unsigned int
: The interrupt ID that was set pending.

GICv3_PPISignalChanged

Fields:

INTERRUPT_ID unsigned int
: The interrupt ID for the PPI.

STATE bool
: The new state of the signal.

GICv3_Redistributor.MemoryMapped_Read

Trace read from a GICv3 memory-mapped register. Fields:

A0 unsigned int
: Affinity level 0 (for banked registers)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

A1 unsigned int
: Affinity level 1 (for banked registers)

A2 unsigned int
: Affinity level 2 (for banked registers)

A3 unsigned int
: Affinity level 3 (for banked registers)

BANKED bool
: Whether the access is to a banked register

NS bool
: Access is non-secure

OFFSET unsigned int
: Offset of address within the GICv3 module

REG_NAME string
: Name of the register being read

VALUE unsigned int
: Value returned

GICv3_Redistributor.MemoryMapped_Read64

Trace read from a GICv3 memory-mapped register. Fields:

A0 unsigned int
: Affinity level 0 (for banked registers)

A1 unsigned int
: Affinity level 1 (for banked registers)

A2 unsigned int
: Affinity level 2 (for banked registers)

A3 unsigned int
: Affinity level 3 (for banked registers)

BANKED bool
: Whether the access is to a banked register

NS bool
: Access is non-secure

OFFSET unsigned int
: Offset of address within the GICv3 module

REG_NAME string
: Name of the register being read

VALUE unsigned int
: Value returned

GICv3_Redistributor.MemoryMapped_Write

Trace write to a GICv3 memory-mapped register. Fields:

A0 unsigned int
: Affinity level 0 (for banked registers)

A1 unsigned int
: Affinity level 1 (for banked registers)

A2 unsigned int
: Affinity level 2 (for banked registers)

A3 unsigned int
: Affinity level 3 (for banked registers)

BANKED bool
: Whether the access is to a banked register

NS bool
: Access is non-secure

OFFSET unsigned int
: Offset of address within the GICv3 module

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

REG_NAME string
: Name of the register being written

UPDATED_VALUE unsigned int
: New value read back from the register; for write-only registers this will be equal to VALUE.

VALUE unsigned int
: Value written

GICv3_Redistributor.MemoryMapped_Write64

Trace write to a GICv3 memory-mapped register. Fields:

A0 unsigned int
: Affinity level 0 (for banked registers)

A1 unsigned int
: Affinity level 1 (for banked registers)

A2 unsigned int
: Affinity level 2 (for banked registers)

A3 unsigned int
: Affinity level 3 (for banked registers)

BANKED bool
: Whether the access is to a banked register

NS bool
: Access is non-secure

OFFSET unsigned int
: Offset of address within the GICv3 module

REG_NAME string
: Name of the register being written

UPDATED_VALUE unsigned int
: New value read back from the register; for write-only registers this will be equal to VALUE.

VALUE unsigned int
: Value written

GICv3_Redistributor.MemoryMapped_WritePort

Trace write to a GICv3 memory-mapped register acting as imp def port. Fields:

FIRST_BYTE_IN_BURST unsigned int
: First byte written in the burst

NS bool
: Access is non-secure

OFFSET unsigned int
: Offset of address within the GICv3 module

REG_NAME string
: Name of the register being written

GICv3_Redistributor_Comms_ReceiveAbove

GICv3 internal communications packet that has been received and is traveling towards the CPU. Fields:

COMMAND enum
: The command type for this packet

DATA unsigned int
: The data (including header) contained within this packet

DATA_LENGTH unsigned int
: Number of bytes in the data (including header)

GICv3_Redistributor_Comms_ReceiveBelow

GICv3 internal communications packet that has been received and is traveling towards the top-level. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

COMMAND enum
: The command type for this packet

DATA unsigned int
: The data (including header) contained within this packet

DATA_LENGTH unsigned int
: Number of bytes in the data (including header)

INDEX unsigned int
: Index for the redistributor at the lower level from which this packet is being sent. Not applicable for sends from CPU interface to RD0.

GICv3_Redistributor_Comms_SendAbove

GICv3 internal communications packet that is being sent towards the top-level. Fields:

COMMAND enum
: The command type for this packet

DATA unsigned int
: The data (including header) contained within this packet

DATA_LENGTH unsigned int
: Number of bytes in the data (including header)

GICv3_Redistributor_Comms_SendBelow

GICv3 internal communications packet that is being sent towards the CPU. Fields:

COMMAND enum
: The command type for this packet

DATA unsigned int
: The data (including header) contained within this packet

DATA_LENGTH unsigned int
: Number of bytes in the data (including header)

INDEX unsigned int
: Index for the redistributor at the lower level from which this packet is being sent. Not applicable for sends from CPU interface to RD0.

GICv3_Redistributor_Internal.MemoryMapped_Read

Trace read from a GICv3 memory-mapped register. Fields:

A0 unsigned int
: Affinity level 0 (for banked registers)

A1 unsigned int
: Affinity level 1 (for banked registers)

A2 unsigned int
: Affinity level 2 (for banked registers)

A3 unsigned int
: Affinity level 3 (for banked registers)

BANKED bool
: Whether the access is to a banked register

NS bool
: Access is non-secure

OFFSET unsigned int
: Offset of address within the GICv3 module

REG_NAME string
: Name of the register being read

VALUE unsigned int
: Value returned

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

GICv3_Redistributor_Internal.MemoryMapped_Read64

Trace read from a GICv3 memory-mapped register. Fields:

A0 unsigned int
: Affinity level 0 (for banked registers)

A1 unsigned int
: Affinity level 1 (for banked registers)

A2 unsigned int
: Affinity level 2 (for banked registers)

A3 unsigned int
: Affinity level 3 (for banked registers)

BANKED bool
: Whether the access is to a banked register

NS bool
: Access is non-secure

OFFSET unsigned int
: Offset of address within the GICv3 module

REG_NAME string
: Name of the register being read

VALUE unsigned int
: Value returned

GICv3_Redistributor_Internal.MemoryMapped_Write

Trace write to a GICv3 memory-mapped register. Fields:

A0 unsigned int
: Affinity level 0 (for banked registers)

A1 unsigned int
: Affinity level 1 (for banked registers)

A2 unsigned int
: Affinity level 2 (for banked registers)

A3 unsigned int
: Affinity level 3 (for banked registers)

BANKED bool
: Whether the access is to a banked register

NS bool
: Access is non-secure

OFFSET unsigned int
: Offset of address within the GICv3 module

REG_NAME string
: Name of the register being written

UPDATED_VALUE unsigned int
: New value read back from the register; for write-only registers this will be equal to VALUE.

VALUE unsigned int
: Value written

GICv3_Redistributor_Internal.MemoryMapped_Write64

Trace write to a GICv3 memory-mapped register. Fields:

A0 unsigned int
: Affinity level 0 (for banked registers)

A1 unsigned int
: Affinity level 1 (for banked registers)

A2 unsigned int
: Affinity level 2 (for banked registers)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

A3 unsigned int
: Affinity level 3 (for banked registers)

BANKED bool
: Whether the access is to a banked register

NS bool
: Access is non-secure

OFFSET unsigned int
: Offset of address within the GICv3 module

REG_NAME string
: Name of the register being written

UPDATED_VALUE unsigned int
: New value read back from the register; for write-only registers this will be equal to VALUE.

VALUE unsigned int
: Value written

GICv3_Redistributor_Internal.MemoryMapped_WritePort

Trace write to a GICv3 memory-mapped register acting as imp def port. Fields:

FIRST_BYTE_IN_BURST unsigned int
: First byte written in the burst

NS bool
: Access is non-secure

OFFSET unsigned int
: Offset of address within the GICv3 module

REG_NAME string
: Name of the register being written

GICv3_RegUpdated32

Fields:

REG_NAME string
: The name of the register being updated.

VALUE unsigned int
: The updated value.

GICv3_RegUpdated64

Fields:

REG_NAME string
: The name of the register being updated.

VALUE unsigned int
: The updated value.

GICv3_TraceWakeRequest

Fields:

INTERFACE unsigned int
: Index of the interface

SETnCLEAR bool
: Whether the signal is set(true) or cleared (false)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.81 GICv3RedistributorInternal - trace

This section describes the trace sources.

ArchMsg.Info.GICv3_DroppedInternalPacket

Fields:

INTERFACE unsigned int
: Index of the interface

ArchMsg.Warning.GICv3_CFGSDISABLE_unsupported

CFGSDISABLE signal has been set but doing so has no effect.

ArchMsg.Warning.GICv3_DroppedInternalPacket

Fields:

INTERFACE unsigned int
: Index of the interface

ArchMsg.Warning.GICv3_ReceivedInvalidCommandFromAbove

Fields:

ACTUAL_LENGTH unsigned int
: Actual length of the data in bytes including the header.

DATA unsigned int
: The data received (including header).

EXPECTED_LENGTH unsigned int
: The length in bytes including header that was expected to be received dependent on command type and potentially the data type in the case of a data write.

TYPE unsigned int
: The command type.

ArchMsg.Warning.GICv3_ReceivedInvalidCommandFromBelow

Fields:

ACTUAL_LENGTH unsigned int
: Actual length of the data in bytes including the header.

DATA unsigned int
: The data received (including header).

EXPECTED_LENGTH unsigned int
: The length in bytes including header that was expected to be received dependent on command type and potentially the data type in the case of a data write.

REDISTRIBUTOR_INDEX unsigned int
: Index for the downstream port that this command was received on.

TYPE unsigned int
: The command type.

ArchMsg.Warning.GICv3_Redistributor.MemoryMapped_ReadIgnored

Fields:

OFFSET unsigned int
: Offset of address within the GICv3 module

ArchMsg.Warning.GICv3_Redistributor.MemoryMapped_ReadReserved

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OFFSET unsigned int
Offset of address within the GICv3 module

ArchMsg.Warning.GICv3_Redistributor.MemoryMapped_ReadWriteOnlyReg

Fields:

OFFSET unsigned int
Offset of address within the GICv3 module
REG_NAME string
Name of the register being read

ArchMsg.Warning.GICv3_Redistributor.MemoryMapped_WriteIgnored

Fields:

OFFSET unsigned int
Offset of address within the GICv3 module

ArchMsg.Warning.GICv3_Redistributor.MemoryMapped_WriteReadOnlyReg

Fields:

OFFSET unsigned int
Offset of address within the GICv3 module
REG_NAME string
Name of the register being written
VALUE unsigned int
Value written

ArchMsg.Warning.GICv3_Redistributor.MemoryMapped_WriteReserved

Fields:

OFFSET unsigned int
Offset of address within the GICv3 module
VALUE unsigned int
Value written

ArchMsg.Warning.GICv3_SPIDropped

Fields:

A0 unsigned int
Affinity level 0 address
A1 unsigned int
Affinity level 1 address
A2 unsigned int
Affinity level 2 address
A3 unsigned int
Affinity level 3 address
INTERRUPT_ID unsigned int
The interrupt ID for the LPI

ArchMsg.Warning.GICv3_WakerequestIgnored

Wakerequest was asserted but the component is not asleep.

GICv3_CFGSDISABLE

Fields:

STATE bool
The new state of the signal.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

GICv3_ClearDropped

Fields:

A0 unsigned int
: Affinity level 0 address

A1 unsigned int
: Affinity level 1 address

A2 unsigned int
: Affinity level 2 address

A3 unsigned int
: Affinity level 3 address

INTERRUPT_ID unsigned int
: The interrupt ID for the LPI

GICv3_Redistributor.MemoryMapped_Read

Trace read from a GICv3 memory-mapped register. Fields:

A0 unsigned int
: Affinity level 0 (for banked registers)

A1 unsigned int
: Affinity level 1 (for banked registers)

A2 unsigned int
: Affinity level 2 (for banked registers)

A3 unsigned int
: Affinity level 3 (for banked registers)

BANKED bool
: Whether the access is to a banked register

NS bool
: Access is non-secure

OFFSET unsigned int
: Offset of address within the GICv3 module

REG_NAME string
: Name of the register being read

VALUE unsigned int
: Value returned

GICv3_Redistributor.MemoryMapped_Read64

Trace read from a GICv3 memory-mapped register. Fields:

A0 unsigned int
: Affinity level 0 (for banked registers)

A1 unsigned int
: Affinity level 1 (for banked registers)

A2 unsigned int
: Affinity level 2 (for banked registers)

A3 unsigned int
: Affinity level 3 (for banked registers)

BANKED bool
: Whether the access is to a banked register

NS bool
: Access is non-secure

OFFSET unsigned int
: Offset of address within the GICv3 module

REG_NAME string
: Name of the register being read

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VALUE unsigned int
Value returned

GICv3_Redistributor.MemoryMapped_Write

Trace write to a GICv3 memory-mapped register. Fields:

A0 unsigned int
Affinity level 0 (for banked registers)
A1 unsigned int
Affinity level 1 (for banked registers)
A2 unsigned int
Affinity level 2 (for banked registers)
A3 unsigned int
Affinity level 3 (for banked registers)
BANKED bool
Whether the access is to a banked register
NS bool
Access is non-secure
OFFSET unsigned int
Offset of address within the GICv3 module
REG_NAME string
Name of the register being written
UPDATED_VALUE unsigned int
New value read back from the register; for write-only registers this will be equal to VALUE.
VALUE unsigned int
Value written

GICv3_Redistributor.MemoryMapped_Write64

Trace write to a GICv3 memory-mapped register. Fields:

A0 unsigned int
Affinity level 0 (for banked registers)
A1 unsigned int
Affinity level 1 (for banked registers)
A2 unsigned int
Affinity level 2 (for banked registers)
A3 unsigned int
Affinity level 3 (for banked registers)
BANKED bool
Whether the access is to a banked register
NS bool
Access is non-secure
OFFSET unsigned int
Offset of address within the GICv3 module
REG_NAME string
Name of the register being written
UPDATED_VALUE unsigned int
New value read back from the register; for write-only registers this will be equal to VALUE.
VALUE unsigned int
Value written

GICv3_Redistributor.MemoryMapped_WritePort

Trace write to a GICv3 memory-mapped register acting as imp def port. Fields:

FIRST_BYTE_IN_BURST unsigned int
First byte written in the burst

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NS bool
: Access is non-secure

OFFSET unsigned int
: Offset of address within the GICv3 module

REG_NAME string
: Name of the register being written

GICv3_Redistributor_Comms_ReceiveAbove

GICv3 internal communications packet that has been received and is traveling towards the CPU. Fields:

COMMAND enum
: The command type for this packet

DATA unsigned int
: The data (including header) contained within this packet

DATA_LENGTH unsigned int
: Number of bytes in the data (including header)

GICv3_Redistributor_Comms_ReceiveBelow

GICv3 internal communications packet that has been received and is traveling towards the top-level. Fields:

COMMAND enum
: The command type for this packet

DATA unsigned int
: The data (including header) contained within this packet

DATA_LENGTH unsigned int
: Number of bytes in the data (including header)

INDEX unsigned int
: Index for the redistributor at the lower level from which this packet is being sent. Not applicable for sends from CPU interface to RD0.

GICv3_Redistributor_Comms_SendAbove

GICv3 internal communications packet that is being sent towards the top-level. Fields:

COMMAND enum
: The command type for this packet

DATA unsigned int
: The data (including header) contained within this packet

DATA_LENGTH unsigned int
: Number of bytes in the data (including header)

GICv3_Redistributor_Comms_SendBelow

GICv3 internal communications packet that is being sent towards the CPU. Fields:

COMMAND enum
: The command type for this packet

DATA unsigned int
: The data (including header) contained within this packet

DATA_LENGTH unsigned int
: Number of bytes in the data (including header)

INDEX unsigned int
: Index for the redistributor at the lower level from which this packet is being sent. Not applicable for sends from CPU interface to RD0.

GICv3_RegUpdated32

Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

REG_NAME `string`
: The name of the register being updated.

VALUE `unsigned int`
: The updated value.

GICv3_RegUpdated64

Fields:

REG_NAME `string`
: The name of the register being updated.

VALUE `unsigned int`
: The updated value.

GICv3_TraceWakeRequest

Fields:

INTERFACE `unsigned int`
: Index of the interface

SETnCLEAR `bool`
: Whether the signal is set(true) or cleared (false)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.82 MMU_400_BASE - trace

This section describes the trace sources.

aarch64_AddressSizeFault_base_address_out_of_range

An AArch64 context got an AddressSizeFault as the base address of the TTBR used is out of range. Fields:

CPAMax unsigned int
: Configured PAMax.

PAMax unsigned int
: Actual PAMax.

context_bank unsigned int
: context_bank.

stage unsigned int
: Stage.

aarch64_AddressSizeFault_in_final_block_address_trace

An AArch64 context got an AddressSizeFault in the final block descriptor. Fields:

CPAMax unsigned int
: Configured PAMax.

PAMax unsigned int
: Actual PAMax.

context_bank unsigned int
: context_bank.

output_address unsigned int
: Output address from stage.

stage unsigned int
: Stage.

aarch64_AddressSizeFault_in_table_descriptor

An AArch64 context got an AddressSizeFault as a descriptor encoded a page and the table base address described was out of range. Fields:

CPAMax unsigned int
: Configured PAMax.

PAMax unsigned int
: Actual PAMax.

context_bank unsigned int
: context_bank.

ipa_address unsigned int
: IPA of descriptor.

ns bool
: Descriptor address is non-secure.

pa_address unsigned int
: PA of descriptor.

stage unsigned int
: Stage.

aarch64_TranslationFault_as_block_too_large

The AArch64 block described by a descriptor is too large. Fields:

context_bank unsigned int
: context_bank.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

level unsigned int
: level.

stage unsigned int
: stage.

aarch64_TranslationFault_input_address_out_of_range

An AArch64 context received an address which was outside of the programmed TxSZ. Fields:

context_bank unsigned int
: context_bank.

input_address unsigned int
: Input address to stage.

ns bool
: Security state of context bank.

stage unsigned int
: Stage.

aarch64_TranslationFault_st2_starting_level_and_tsize_inconsistent

An AArch64 context got a TranslationFault as the specified starting level and TxSZ field are inconsistent. Fields:

context_bank unsigned int
: context_bank.

stage unsigned int
: Stage.

add_ns_context_fault

A context fault occurred and this is the record it wishes to generate. Fields:

SMMU_CBFRSYNRAn unsigned int
: SMMU_CBFRSYNRAn value that it wishes to write.

SMMU_CBn_FAR unsigned int
: SMMU_CBn_FAR value that it wishes to write.

SMMU_CBn_FSR unsigned int
: SMMU_CBn_FSR value that it wishes to write.

SMMU_CBn_FSYNR0 unsigned int
: SMMU_CBn_FSYNR0 value that it wishes to write.

SMMU_CBn_FSYNR1 unsigned int
: SMMU_CBn_FSYNR1 value that it wishes to write.

SMMU_CBn_IPAFAR unsigned int
: (SMMUv2 only) SMMU_CBn_IPAFAR value that it wishes to write. The value is UNKNOWN if this is a stage 1 context.

context_bank unsigned int
: Context bank.

going_to_be_lost bool
: This fault record is going to be lost as there is already an unhandled fault record.

add_ns_global_fault

Attempt to raise a Non-Secure global fault. Fields:

SMMU_GFAR unsigned int
: The fault address that we are attempting to record.

SMMU_GFSR unsigned int
: The bits that we are attempting to set in the SMMU_GFSR.

SMMU_GFSYNR0 unsigned int
: The syndrome we are attempting to record.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SMMU_GFSYNR1 unsigned int
: The syndrome we are attempting to record.

going_to_be_lost bool
: This fault record is going to be lost as there is already an unhandled fault record.

add_s_context_fault

A context fault occurred and this is the record it wishes to generate. Fields:

SMMU_CBFRSYNRAn unsigned int
: SMMU_CBFRSYNRAn value that it wishes to write.

SMMU_CBn_FAR unsigned int
: SMMU_CBn_FAR value that it wishes to write.

SMMU_CBn_FSR unsigned int
: SMMU_CBn_FSR value that it wishes to write.

SMMU_CBn_FSYNR0 unsigned int
: SMMU_CBn_FSYNR0 value that it wishes to write.

SMMU_CBn_FSYNR1 unsigned int
: SMMU_CBn_FSYNR1 value that it wishes to write.

SMMU_CBn_IPAFAR unsigned int
: (SMMUv2) SMMU_CBn_IPAFAR value that it wishes to read. This becomes UNKNOWN as this is a secure bank.

context_bank unsigned int
: Context bank.

going_to_be_lost bool
: This fault record is going to be lost as there is already an unhandled fault record.

add_s_global_fault

Attempt to raise a Secure global fault. Fields:

SMMU_SGFAR unsigned int
: The fault address that we are attempting to record.

SMMU_SGFSR unsigned int
: The bits that we are attempting to set in the SMMU_SGFSR.

SMMU_SGFSYNR0 unsigned int
: The syndrome we are attempting to record.

SMMU_SGFSYNR1 unsigned int
: The syndrome we are attempting to record.

going_to_be_lost bool
: This fault record is going to be lost as there is already an unhandled fault record.

ats_cb_atsr_read

A context bank Address Translation Status Register read occurred (Writes are reported through ats_cb_operation_doesnt_exist_trace). Fields:

ACTIVE bool
: The ACTIVE bit returned.

context_id unsigned int
: Context bank ID.

is_debug bool
: The access was a debug access.

ats_cb_operation_doesnt_exist

An attempt was made to use a Context Bank Address Translation Operation that didn't exist for some reason. Fields:

context_id unsigned int
: Context bank ID.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

desc string
: Description of why it doesn't exist.

ats_cb_operation_queued

A new Context Bank Address Translation Operation was queued. Fields:

VA unsigned int
: Virtual Address.

ats_is_privileged bool
: The ATS operation is asking for a 'privileged' access.

ats_is_read bool
: The ATS operation is asking for a 'read' access.

context_id unsigned int
: Context bank ID.

is_debug bool
: Is for a debug request.

ats_cb_operation_received_while_ats_in_progress

Another Context Bank Address Translation operation was received whilst one was currently in progress. This is UNPREDICTABLE. Fields:

context_id unsigned int
: Context bank ID.

desc string
: Description of new ATS operation.

ats_global_atsr_read

A Global Address Translation Status Register read occurred (Writes are reported through ats_cb_operation_doesnt_exist_trace). Fields:

ACTIVE bool
: The ACTIVE bit returned.

is_debug bool
: The access was a debug access.

which enum
: Which global ATSR register?

ats_global_operation_doesnt_exist

An attempt was made to use a Context Bank Address Translation Operation that didn't exist for some reason. Fields:

desc string
: Description of why it doesn't exist.

is_non_secure enum
: Is the operation non-secure.

ats_global_operation_queued

A new Context Bank Address Translation Operation was queued. Fields:

VA unsigned int
: Virtual Address.

ats_is_privileged bool
: The ATS operation is asking for a 'privileged' access.

ats_is_read bool
: The ATS operation is asking for a 'read' access.

context_id unsigned int
: Context bank ID.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

is_debug bool
: Is for a debug request.

is_non_secure enum
: Is the operation non-secure.

is_s12 bool
: Is a stage 1 and stage 2 translation.

ats_global_operation_received_while_ats_in_progress

Another Global Address Translation operation was received whilst one was currently in progress. This is UNPREDICTABLE. Fields:

context_id unsigned int
: Context bank ID of new operation.

desc string
: Description of new ATS operation.

is_non_secure enum
: Is the operation non-secure.

ats_op_started

A queued address translation operation has started. Fields:

VA unsigned int
: Input virtual address.

context_bank unsigned int
: Context bank if which == 'context_bank'.

is_debug bool
: The operation is a debug ATS command.

is_privileged bool
: Is a privileged operation.

is_read bool
: Is a read operation.

is_s12 bool
: Is for a stage 1 and 2 translation.

which enum
: Which group of operations?

cfg_cttw

The cfg_cttw signal level that indicates that the SoC support coherent page walks. Fields:

value bool
: The signal level.

cfg_flt_irpt

Configuration Fault Interrupt. Fields:

asserted bool
: The signal is asserted.

is_non_secure bool
: cfg_flt_irpt_ns or cfg_flt_irpt_s.

changed_PAGESIZE

The PAGESIZE has been told to change. Fields:

is_64KiB bool
: The PAGESIZE is 64 KiB.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

changing_security_state_of_context_bank

The specified bank has changed its security state. Fields:

context_bank unsigned int
: Context bank number.

now_is_non_secure bool
: The context bank is now non-secure.

prev_is_non_secure bool
: Prior to this change in the SMMU_SCR1.NSNUMCBO then the bank was non-secure.

cmo_permission_fault

A Cache Maintenance Operation (CMO) generated a permission fault. Fields:

abort_returned_to_device bool
: An abort is returned to the device.

cmo enum
: Cache Maintenance Operation.

context_id unsigned int
: Context ID that receives the fault.

input_address unsigned int
: The address of the CMO.

cmo_permission_model_being_applied

The cache maintenance operation permission model is being applied. Fields:

aarch64 bool
: Is the initial context bank aarch64?

fault_action enum
: Whether there is a fault or not.

incoming_cmo_operation enum
: The CMO operation.

instruction bool
: Is the CMO instruction?

outgoing_cmo_operation enum
: The CMO operation.

privileged bool
: Is the CMO privileged?

stage1_perms enum
: The permissions at stage 1.

stage2_perms enum
: The permissions at stage 1.

comb_irpt

Combined interrupt signal. This is the OR of all of the non-secure interrupt signals. Fields:

asserted bool
: The signal is asserted.

is_non_secure bool
: Is the comb_irpt_ns signal or the comb_irpt_s signal.

configuration_access_fault_unimplemented_stream_mapping_register_group

This occurs when a configuration transaction attempts to access a stream mapping register group that doesn't exist and if the implementation treats this as a Configuration Access Fault. Fields:

address unsigned int
: Address that was used.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

group_number unsigned int
: Group number if it were to exist/be accessible.

is_non_secure bool
: Is the configuration access non-secure?

is_read bool
: Is the configuration access a read?

context_bank_commentary

These messages are verbose commentary about what the context bank is doing. Fields:

output string
: The stream output.

context_bank_mmu_off_stage1

This is the transformation applied by the stage 1 SMMU_CBn_SCTLR. Fields:

SMMU_CBn_SCTLR unsigned int
: Value of the SMMU_CBn_SCTLR used.

context_id unsigned int
: Context bank id the transaction has been mapped to.

trans_id unsigned int
: Transaction ID.

context_bank_mmu_off_stage2

This is the transformation applied by the stage 2 SMMU_CBn_SCTLR. Fields:

SMMU_CBn_SCTLR unsigned int
: Value of the SMMU_CBn_SCTLR used.

context_id unsigned int
: Context bank id the transaction has been mapped to.

trans_id unsigned int
: Transaction ID.

ctx_irpt_ns

Context Interrupt, this can only ever be non-secure. Fields:

asserted bool
: The signal is asserted.

debug

This is the textual output of the debug messages of the SMMU model, this need not necessarily make much sense to the user, but is useful for sending to the model writers when reporting bugs. Fields:

output string
: The stream output.

denied_access_due_to_imp_def_reason

The register access was denied because of some IMPLEMENTATION DEFINED reason due to the transaction attributes. Fields:

address unsigned int
: Address of register access.

desc string
: Description of register.

denied_access_due_to_smmu_scr1_gasrae

The register access was denied because SMMU_SCR1.GASRAE has locked down the register. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

address unsigned int
: Address of register access.

dvm_tlbinvalidate_complete

The DVM TLB Invalidate message completed.

dvm_tlbinvalidate_received

A DVM message for a TLB Invalidate has been received. Fields:

address unsigned int
: The VA or IPA to use if match_address.

asid unsigned int
: The ASID to match if match_asid.

by_ipa bool
: The operation is for an IPA operation if match_address.

ignored enum
: The DVM message was ignored.

last_level bool
: The operation is for last level if supported.

match_address bool
: Match the address field.

match_asid bool
: Match the asid field.

match_vmid bool
: Match the vmid field.

prot enum
: The protection level for which this TLB Invalidate will operate on.

security_world enum
: The security world that this will apply to.

stage1_only bool
: The operation is for stage 1 only if supported.

vmid unsigned int
: The VMID to match if match_vmid.

end_stall_transaction

Indicates a transaction unstalling. Fields:

context_id unsigned int
: The context id that we are stalling against.

trans_id unsigned int
: Transaction id of replaying transaction.

unstall_state enum
: Describes why we unstalled. If we are head of line then we are either replay/terminate, otherwise we will be not be.

error

These messages are about activity occurring on the SMMU that is considered an error. Fields:

output string
: The stream output.

error_contiguous_bit_is_bad

The contiguous bit is inconsistent with the descriptor. Fields:

B_mask unsigned int
: The mask of the address that is zapped if we were to use the contiguous bit.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

aarch64_contig_bit_too_high bool
: AArch64 claims that the contig bit can't be set for this particular set of start levels and input address size.

input_address unsigned int
: The input address that will be combined with the output address.

ln2_size_in_bytes_of_region_if_contiguous unsigned int
: The number of address bits that form the 'offset' within the contiguous region.

out_of_range_of_input bool
: The range mapped by a contiguous bit being on is larger than the input range for this stage.

output_address unsigned int
: The output address as contained in the descriptor.

output_address_if_contig_bit_uses_input_address unsigned int
: Output address if contig bit was fully taken (maximally sized region) and those bits came from the input address.

output_address_inconsistent bool
: The output address in the descriptor is inconsistent with its position in the table.

stage_and_level unsigned int
: Top 8 bits are the stage, bottom 8 bits are level.

error_invalid_smmu_cbar

A SMMU_CBAR has been encountered for a secure stream that is not of the Stage 1 with Stage 2 bypass form. Fields:

SMMU_CBARn unsigned int
: The corresponding SMMU_CBARn for the context.

action enum
: Action taken.

context_id unsigned int
: Context ID.

trans_id unsigned int
: Transaction id.

error_tlb_entries_overlap

A TLB entry was attempted to be inserted into the TLB and was determined that it overlaps an existing entry. This check is not perfect but will catch simple errors. Fields:

end_address_of_new_entry unsigned int
: End address of new entry.

end_address_of_old_entry unsigned int
: End address of old entry.

index_of_new_entry unsigned int
: Index of new entry.

index_of_old_entry unsigned int
: Index of old entry.

start_address_of_new_entry unsigned int
: Start address of new entry.

start_address_of_old_entry unsigned int
: Start address of old entry.

extended_attributes_for_stage

This gives the attributes that a particular stage gives prior to being applied to the incoming attributes. Fields:

inner enum
: Inner cacheability.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ns bool
: NS/S security state.

outer enum
: Outer cacheability.

racfg enum
: RACFG from descriptor (if exists).

shareability enum
: Shareability.

stage unsigned int
: Stage that this record is for.

trans_id unsigned int
: Transaction ID this is for.

wacfg enum
: WACFG from descriptor (if exists).

glbl_flt_irpt

Global Fault Interrupt. Fields:

asserted bool
: The signal is asserted.

is_non_secure bool
: glbl_flt_irpt_ns or glbl_flt_irpt_s.

imp_def_non_SMMU_CBn_RESUME_register_access_transaction_forces_replay

There is an IMP DEF option to replay stalled transactions in affected banks. This records that the implementation has made use of this option. Fields:

context_id unsigned int
: Context ID of affected bank.

why enum
: Why did this replay occur?

implementation_postprocess

This describes the non-architectural transformation that the implementation makes. Fields:

end_adomain enum
: End adomain value.

end_inner_acache enum
: End inner acache value.

end_outer_acache enum
: End outer acache value.

is_read bool
: Is a read transaction.

start_adomain enum
: Start adomain value.

start_inner_acache enum
: Start inner acache value.

start_outer_acache enum
: Start outer acache value.

info

These are information messages about what is happening in the SMMU. Fields:

output string
: The stream output.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

invalid_address_range

An input or output address range check failed. Input address range checks are reported as TranslationFault, output ones as AddressSizeFault. Fields:

context_bank unsigned int
: Context bank.

offending_address unsigned int
: The address that failed the address check.

why enum
: The reason why the address was invalid.

ld_access_fault

An Access Fault occurred. Fields:

context_bank unsigned int
: Context bank.

input_address unsigned int
: The input address used for this descriptor.

is_non_secure bool
: Is the input address non-secure?

level unsigned int
: Level of walk.

stage unsigned int
: Stage of walk.

ld_descriptor_fetch_failed

Long Descriptor AArch32 or AArch64, level 'n' descriptor fetch failed. Fields:

context_bank unsigned int
: Context bank.

ipa_address unsigned int
: IPA of the descriptor address.

is_non_secure bool
: Is the page walk and input addresses non-secure?

level unsigned int
: Level of walk.

pa_address unsigned int
: PA of the descriptor address.

stage unsigned int
: Stage of walk.

ld_descriptor_invalid

Long Descriptor AArch32 or AArch64 encodes an invalid entry. Fields:

context_bank unsigned int
: Context bank.

ipa_address unsigned int
: IPA of the descriptor address.

level unsigned int
: Level of walk.

pa_address unsigned int
: PA of the descriptor address.

ssd_ns bool
: Is the transaction classified as non-secure.

stage unsigned int
: Stage of walk.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ld_hyp_mode_address_out_of_range

A hypervisor input address is out of range of T0SZ. Fields:

context_bank unsigned int
: The context bank.

input_address unsigned int
: The VA input address used for this descriptor.

is_non_secure bool
: Is this for the Non-secure side?

ld_ipa_out_of_range

A stage 2 context bank was given an address that didn't match T0SZ. Fields:

context_bank unsigned int
: The context bank.

input_address_ipa unsigned int
: The IPA input address used for this descriptor.

is_non_secure bool
: Is this for the Non-secure side?

ld_ipa_out_of_range_for_st1ptw_st2_mmu_off

(SMMUv2) The stage 2 context (nested with MMU off or bypass) for a stage 1 translation had an address that was out of the allowed range. Fields:

context_bank unsigned int
: The context bank.

input_address_ipa unsigned int
: The IPA input address used for this descriptor.

is_non_secure bool
: Is this for the Non-secure side?

ld_last_level_descriptor_does_not_encode_a_page

Long Descriptor, last level descriptor must encode a page but didn't. Fields:

context_bank unsigned int
: Context bank.

ipa_address unsigned int
: IPA of the descriptor address.

level unsigned int
: Level of walk.

pa_address unsigned int
: PA of the descriptor address.

ssd_ns bool
: Is the transaction classified as non-secure?

stage unsigned int
: Stage of walk.

ld_misprogramming_fault_trace

A misprogramming occurred that the implementation faulted. Fields:

context_bank unsigned int
: Context bank.

input_address unsigned int
: Input address.

stage unsigned int
: Stage.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

why enum
: Why the fault occurred.

ld_st2_produced_any_device_memory_for_st1ptw

A stage 2 translation process for a stage 1 page walk address produce a any-device memory type. Fields:

context_bank unsigned int
: Context bank.

input_address unsigned int
: The input address used for this descriptor.

level unsigned int
: Level of walk.

ld_st2_s1ptw_has_no_access_rights

A stage 2 translation process for a stage 1 page walk address produce a permission fault. Fields:

context_bank unsigned int
: Context bank.

input_address unsigned int
: The input address used for this descriptor.

level unsigned int
: Level of walk.

ld_stage1_non_hyp_no_ttbr_match

A stage 1 non-hyp entry failed to match TTBR0 or TTBR1. Fields:

context_bank unsigned int
: The context bank.

input_address unsigned int
: The input address used.

is_non_secure bool
: Is this for the Non-secure side?

ld_walk_process_is_disabled

A L1 or L2 short descriptor access flag has disabled access for this (super-)section/page. Fields:

context_bank unsigned int
: The context bank.

input_address unsigned int
: The input address used.

is_non_secure bool
: Is this for the Non-secure side?

stage unsigned int
: Which change this corresponds to.

ttbr_id unsigned int
: Which TTBR we are using (is 0 for a stage 2).

non_PTW_external_fault_recorded

A non Page Table Walk access was recorded when a translated transaction went downstream, but the downstream system aborted it. This is recorded as an asynchronous fault. In the model, we can provide more information about the faulting transaction. Fields:

downstream_address unsigned int
: The downstream address.

downstream_ns enum
: The downstream NS state. This is not the SSD.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

is_read enum
: Is the transaction a read?

result_reported_upstream enum
: The result that is reported upstream.

sGFSYNR0 unsigned int
: The value of sGFSYNR0 that will attempt to be recorded (all 1s mean it will not be updated).

sGFSYNR1 unsigned int
: The value of sGFSYNR1 that will attempt to be recorded (all 1s mean it will not be updated).

smmu_scr1_gefro bool
: The value of SMMU_SCR1.GEFRO.

smmu_xcr0_gfre bool
: The appropriate SMMU_sCR0.GFRE bit. If this is 0 then it squash the downstream abort.

ssd_ns enum
: SSD.

upstream_address unsigned int
: The upstream address.

upstream_ns enum
: The upstream NS state. This is not the SSD.

post_s2crn_attributes

The attributes after the SMMU_S2CRn transformation has been applied. Fields:

adomain enum
: The actual attributes of the access that was used, after IMP DEF mangling.

aprot enum
: The actual attributes of the access that was used, after IMP DEF mangling.

inner_cache enum
: The actual attributes of the access that was used, after IMP DEF mangling.

inner_transient bool
: Is the transaction marked with the transient hint?

outer_cache enum
: The actual attributes of the access that was used, after IMP DEF mangling.

outer_transient bool
: Is the transaction marked with the transient hint?

trans_id unsigned int
: Transaction id.

v8_dev_gathering bool
: Is the transaction marked with the V8 Device Gathering property?

v8_dev_reordering bool
: Is the transaction marked with the V8 Device Reordering property?

ptw_read

Page Table Walk (read). This is the physical access that the SMMU is making. Fields:

adomain enum
: The actual attributes of the access that was used, after IMP DEF mangling.

aprot enum
: The actual attributes of the access that was used, after IMP DEF mangling.

auser unsigned int
: The AUSER fields.

data unsigned int
: The data read in little-endian format.

hyp bool
: Current mode is hyp.

inner_cache enum
: The actual attributes of the access that was used, after IMP DEF mangling.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

input_address unsigned int
: Input address this lookup is for.

ns bool
: This is for SSD Non-secure.

outer_cache enum
: The actual attributes of the access that was used, after IMP DEF mangling.

pa_address unsigned int
: Physical address.

result enum
: The PTW result.

st2_st1_context_banks unsigned int
: The stage 1 and 2 context bank that this walk is for (if there is one, otherwise 0xFF). Stage 1 is the lowest byte, stage 2 the highest.

stage_and_level unsigned int
: Stage and level that this walk is for. Top byte is stage, bottom is level.

trans_id unsigned int
: Transaction id.

ptw_read_st1_invalid_long_descriptor

Page Table Walk (read). The descriptor that the PTW fetched is invalid. Fields:

hyp bool
: Current mode is hyp.

input_address unsigned int
: Input address this lookup is for.

level unsigned int
: The page table level this walk is for.

ns bool
: This is for SSD Non-secure.

pa_address unsigned int
: Physical address.

st1_context_bank unsigned int
: The stage 1 context bank that this walk is for (if there is one, otherwise 0xFF).

st2_context_bank unsigned int
: The stage 2 context bank that this walk is for (if there is one, otherwise 0xFF).

trans_id unsigned int
: Transaction id.

ptw_read_st1_leaf_long_descriptor

Page Table Walk (read). The descriptor that the PTW fetched is a block or page and this is the decode. Fields:

AF bool
: Access Flag.

AP21 enum
: The access permissions.

AttrIndx210 unsigned int
: The attribute index into the MAIR0/1.

NS bool
: The encoding is for non-secure if this is a secure fetch.

PXN bool
: Privileged eXecute Never.

SH10 enum
: The shareability.

XN bool
: eXecute Never.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

contiguous bool
: Contiguous hint.

hyp bool
: Current mode is hyp.

input_address unsigned int
: Input address this lookup is for.

level unsigned int
: The page table level this walk is for.

mair unsigned int
: The MAIRn encoding attribute.

nG bool
: not Global.

ns bool
: The agent is Non-Secure.

output_address unsigned int
: Output address.

pa_address unsigned int
: Physical address.

st1_context_bank unsigned int
: The stage 1 context bank that this walk is for (if there is one, otherwise 0xFF).

st2_context_bank unsigned int
: The stage 2 context bank that this walk is for (if there is one, otherwise 0xFF).

trans_id unsigned int
: Transaction id.

ptw_read_st1_table_long_descriptor

Page Table Walk (read). The descriptor that the PTW fetched is a Table, this decodes the fields. Fields:

APTable enum
: Remove permissions independently of subsequent descriptors.

NSTable bool
: The next level table descriptor is forced to non-secure.

PXNTable bool
: Force PXN independently of subsequent descriptors.

TableAddress unsigned int
: Address of the next table.

XNTable bool
: Force XN independently of subsequent descriptors.

hyp bool
: Current mode is hyp.

input_address unsigned int
: Input address this lookup is for.

level unsigned int
: The page table level this walk is for.

ns bool
: This is for SSD Non-secure.

pa_address unsigned int
: Physical address.

st1_context_bank unsigned int
: The stage 1 context bank that this walk is for (if there is one, otherwise 0xFF).

st2_context_bank unsigned int
: The stage 2 context bank that this walk is for (if there is one, otherwise 0xFF).

trans_id unsigned int
: Transaction id.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ptw_read_st1_v7s_fault_descriptor

Page Table Walk (read). The descriptor encodes a fault descriptor. Fields:

level unsigned int
: The level of the page table.

trans_id unsigned int
: Transaction id.

ptw_read_st1_v7s_large_page_descriptor

Page Table Walk (read). The descriptor encodes a large page descriptor. Fields:

AP210 unsigned int
: AP[2:0].

S bool
: Shareable.

TEX210CB unsigned int
: {TEX[2:0],C,B}.

XN bool
: eXecute Never.

large_page_base_address unsigned int
: Base address of the large page.

nG bool
: not Global.

trans_id unsigned int
: Transaction id.

ptw_read_st1_v7s_page_table_descriptor

Page Table Walk (read). The descriptor encodes a page table descriptor pointing to an L2 table. Fields:

NS bool
: Encodes a non-secure entry.

PXN bool
: Privileged eXecute Never.

domain unsigned int
: Domain.

l2_table_address unsigned int
: L2 page table address.

trans_id unsigned int
: Transaction id.

ptw_read_st1_v7s_section_descriptor

Page Table Walk (read). The descriptor encodes a section descriptor. Fields:

AP210 unsigned int
: AP[2:0].

NS bool
: Encodes a non-secure entry.

PXN bool
: Privileged eXecute Never.

S bool
: Shareable.

TEX210CB unsigned int
: {TEX[2:0],C,B}.

XN bool
: eXecute Never.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

domain unsigned int
: Domain.

nG bool
: not Global.

section_base_address unsigned int
: Base address of section.

trans_id unsigned int
: Transaction id.

ptw_read_st1_v7s_small_page_descriptor

Page Table Walk (read). The descriptor encodes a small page descriptor. Fields:

AP210 unsigned int
: AP[2:0].

S bool
: Shareable.

TEX210CB unsigned int
: {TEX[2:0],C,B}.

XN bool
: eXecute Never.

nG bool
: not Global.

small_page_base_address unsigned int
: Base address of the small page.

trans_id unsigned int
: Transaction id.

ptw_read_st1_v7s_super_section_descriptor

Page Table Walk (read). The descriptor encodes a section descriptor. Fields:

AP210 unsigned int
: AP[2:0].

NS bool
: Encodes a non-secure entry.

PXN bool
: Privileged eXecute Never.

S bool
: Shareable.

TEX210CB unsigned int
: {TEX[2:0],C,B}.

XN bool
: eXecute Never.

nG bool
: not Global.

super_section_base_address unsigned int
: Base address of the super section.

trans_id unsigned int
: Transaction id.

ptw_read_st2_invalid_descriptor

Page Table Walk (read). The descriptor that the PTW fetched is invalid. Fields:

hyp bool
: Current mode is hyp.

input_address unsigned int
: Input address this lookup is for.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

level unsigned int
: The page table level this walk is for.

ns bool
: This is for SSD Non-secure.

pa_address unsigned int
: Physical address.

st1_context_bank unsigned int
: The stage 1 context bank that this walk is for (if there is one, otherwise 0xFF).

st2_context_bank unsigned int
: The stage 2 context bank that this walk is for (if there is one, otherwise 0xFF).

trans_id unsigned int
: Transaction id.

ptw_read_st2_leaf_descriptor

Page Table Walk (read). The descriptor that the PTW fetched is a block or page and this is the decode. Fields:

AF bool
: Access Flag.

HAP21 enum
: The access permissions.

MemAttr3_0 enum
: The memory attributes.

RACFG enum
: The extra SMMU RACFG field.

SH10 enum
: The shareability.

WACFG enum
: The extra SMMU WACFG field.

XN bool
: eXecute Never.

contiguous bool
: Contiguous hint.

hyp bool
: Current mode is hyp.

input_address unsigned int
: Input address this lookup is for.

level unsigned int
: The page table level this walk is for.

ns bool
: This is for SSD Non-secure.

output_address unsigned int
: Output address.

pa_address unsigned int
: Physical address.

st1_context_bank unsigned int
: The stage 1 context bank that this walk is for (if there is one, otherwise 0xFF).

st2_context_bank unsigned int
: The stage 2 context bank that this walk is for (if there is one, otherwise 0xFF).

trans_id unsigned int
: Transaction id.

ptw_read_st2_table_descriptor

Page Table Walk (read). The descriptor that the PTW fetched is a Table, this decodes the fields. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

APTable enum
: Remove permissions independently of subsequent descriptors.

NSTable bool
: The next level table descriptor is forced to non-secure.

PXNTable bool
: Force PXN independently of subsequent descriptors.

TableAddress unsigned int
: Address of the next table.

XNTable bool
: Force XN independently of subsequent descriptors.

hyp bool
: Current mode is hyp.

input_address unsigned int
: Input address this lookup is for.

level unsigned int
: The page table level this walk is for.

ns bool
: This is for SSD Non-secure.

pa_address unsigned int
: Physical address.

st1_context_bank unsigned int
: The stage 1 context bank that this walk is for (if there is one, otherwise 0xFF).

st2_context_bank unsigned int
: The stage 2 context bank that this walk is for (if there is one, otherwise 0xFF).

trans_id unsigned int
: Transaction id.

raw_want_to_change_context_interrupt

This represents a context's bank intent to want to change an interrupt pin. However, this doesn't take into account the accessibility of the bank, nor if other banks are driving the pin. Fields:

accessible bool
: Is the interrupt ID accessible to this context bank?

asserted bool
: Is this context bank attempting to assert the interrupt?

context_bank unsigned int
: Context bank attempting to change the interrupt pin.

interrupt_id unsigned int
: The interrupt ID it is attempting to change.

register_disallowed_read_string

A text representation of the read of a register that was disallowed. Fields:

out string
: The text description of the register value read

register_disallowed_write_string

A text representation of the write of a register write that was disallowed. Fields:

in string
: The text description of the register value written

register_read

Read of the register file. Fields:

address unsigned int
: The address of the register.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

context_id unsigned int
: Context id bank (or 0xFF if not a context bank)

ln2_width enum
: The log2 width of the access

ns enum
: The security state of the transaction addressing the register

register_trans_id unsigned int
: Register transaction ID (if top bit set then a debug transaction)

result enum
: The result of the access

value unsigned int
: The byte-stream of all the data (little-endian)

register_read_reserved

A text representation of an access to a register address that is reserved. Fields:

in string
: The text description of the register value

register_read_string

A text representation of the read of a register. Fields:

out string
: The text description of the register value read

register_write

Write to the register file. Fields:

address unsigned int
: The address of the register.

context_id unsigned int
: Context id bank (or 0xFF if not a context bank)

ln2_width enum
: The log2 width of the access

ns enum
: The security state of the transaction addressing the register

register_trans_id unsigned int
: Register transaction ID (if top bit set then a debug transaction)

result enum
: The result of the access

value unsigned int
: The byte-stream of all the data (little-endian)

register_write_reserved

A text representation of an access to a register address that is reserved. Fields:

in string
: The text description of the register value

register_write_string

A text representation of the write of a register. Fields:

in string
: The text description of the register value written

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

reset_sequence

Marks the beginning and end of the reset sequence. Fields:

starting bool
: True if starting the reset sequence, false when finished.

signal_reset

The reset signal level. Fields:

value bool
: The signal level.

smmu_faulting_transaction

This is a trace of transaction that are not expected to fault. This does not include those transaction that are faulting by stage 2 or where the access flag or permission model is not allowing the transaction. Fields:

context_id unsigned int
: The context id of the transaction used, if we failed to immediately hit in the TLB.

input_address unsigned int
: The input address of the transaction group.

mapped_attributes unsigned int
: The mapped attributes of the transaction.

original_attributes unsigned int
: The original unmapped attributes of the transaction.

output_address unsigned int
: The output address of the transaction group.

ssd_index unsigned int
: The Security State Determination Index if applicable.

ssd_ns enum
: The Security State Determination of the transaction.

state enum
: The state of the transaction.

stream_id unsigned int
: The stream id of the transaction

tbu unsigned int
: Translation Buffer Unit number.

tlb_entry_index_st1 unsigned int
: If this was mapped by a TLB entry index then this is the one used for the stage 1 translation.

tlb_entry_index_st2 unsigned int
: If this was mapped by a TLB entry index then this is the one used for the stage 2 translation.

trans_id unsigned int
: Transaction id.

type_of_transaction enum
: The type of transaction.

smmu_final_transaction

This is a trace of the SMMU transaction after it has been fully remapped. Fields:

context_id unsigned int
: The context id of the transaction used, if we failed to immediately hit in the TLB.

input_address unsigned int
: The input address of the transaction group.

mapped_attributes unsigned int
: The mapped attributes of the transaction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

original_attributes unsigned int
: The original unmapped attributes of the transaction.

output_address unsigned int
: The output address of the transaction group.

ssd_index unsigned int
: The Security State Determination Index if applicable.

ssd_ns enum
: The Security State Determination of the transaction.

state enum
: The state of the transaction.

stream_id unsigned int
: The stream id of the transaction

tbu unsigned int
: Translation Buffer Unit number.

tlb_entry_index_st1 unsigned int
: If this was mapped by a TLB entry index then this is the one used for the stage 1 translation.

tlb_entry_index_st2 unsigned int
: If this was mapped by a TLB entry index then this is the one used for the stage 2 translation.

trans_id unsigned int
: Transaction id.

type_of_transaction enum
: The type of transaction.

smmu_initial_transaction

This is the initial transaction group request to remap. This represents a bundle of transactions with the same attributes but different addresses within a certain range around the address. Fields:

input_address unsigned int
: The input address of the transaction group.

kind_addressed enum
: The kind of transaction this represents if it is an addressed transaction

original_attributes unsigned int
: The original unmapped attributes of the transaction.

ssd_index unsigned int
: The Security State Determination Index if applicable.

ssd_ns enum
: The Security State Determination of the transaction.

stream_id unsigned int
: The stream id of the transaction.

tbu unsigned int
: Translation Buffer Unit number.

trans_id unsigned int
: Transaction id.

type_of_transaction enum
: The type of transaction.

smmu_iprt_xgirpt

This trace source records the state of the SMMU_gIrpt/SMMU_NSgIrpt pins. Fields:

asserted bool
: The interrupt is now asserted.

is_non_secure bool
: Identifies the pin as SMMU_NSgIrpt.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

smmu_irpt_combined_context_interrupt_pin

This trace source records when there is at least one context bank in the specific world with an interrupt pin raised. Fields:

asserted bool
: The interrupt is now asserted.

ns bool
: Non-secure

smmu_irpt_context_interrupt_pin

This trace source records when a context bank interrupt pin changes state. Fields:

asserted bool
: The interrupt is now asserted.

interrupt_pin unsigned int
: Interrupt pin number.

smmu_irpt_context_interrupt_pin_drivers

This trace source records when a context bank tries to assert an interrupt. Multiple banks can drive the same pin and so long as one is driving it then it is asserted. Fields:

context_bank unsigned int
: The context bank number changing its request for the pin

interrupt_pin unsigned int
: Interrupt pin number.

new_count_asserting_pin unsigned int
: The number of banks asserting the pin

this_context_asserting bool
: The request that this context is making -- if this is true then the context bank is asserting a request

smmu_irpt_xgcfgirpt

This trace source records the state of the SMMU_gCfgIrpt/SMMU_NSgCfgIrpt pins. Fields:

asserted bool
: The interrupt is now asserted.

is_non_secure bool
: Identifies the pin as SMMU_NSgCfgIrpt.

smmu_st1_st2fault_transaction

This is a trace of transactions that map to stage 1 with stage 2 faulting. Fields:

context_id unsigned int
: The context id of the transaction used, if we failed to immediately hit in the TLB.

input_address unsigned int
: The input address of the transaction group.

mapped_attributes unsigned int
: The mapped attributes of the transaction.

original_attributes unsigned int
: The original unmapped attributes of the transaction.

output_address unsigned int
: The output address of the transaction group.

ssd_index unsigned int
: The Security State Determination Index if applicable.

ssd_ns enum
: The Security State Determination of the transaction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

state enum
: The state of the transaction.

stream_id unsigned int
: The stream id of the transaction

tbu unsigned int
: Translation Buffer Unit number.

tlb_entry_index_st1 unsigned int
: If this was mapped by a TLB entry index then this is the one used for the stage 1 translation.

tlb_entry_index_st2 unsigned int
: If this was mapped by a TLB entry index then this is the one used for the stage 2 translation.

trans_id unsigned int
: Transaction id.

type_of_transaction enum
: The type of transaction.

smrg_matched

Stream Match Register Group matched. Fields:

desc string
: A textual description of the register group

n unsigned int
: The SMRG that we matched against.

trans_id unsigned int
: The transaction id that we are matching against.

start_ptw_read

Page Table Walk (read). This is the start of the physical access that the SMMU is making. Fields:

adomain enum
: The actual attributes of the access that was used, after IMP DEF mangling.

aprot enum
: The actual attributes of the access that was used, after IMP DEF mangling.

auser unsigned int
: The AUSER fields.

hyp bool
: Current mode is hyp.

inner_cache enum
: The actual attributes of the access that was used, after IMP DEF mangling.

input_address unsigned int
: Input address this lookup is for.

ns bool
: This is for SSD Non-secure.

outer_cache enum
: The actual attributes of the access that was used, after IMP DEF mangling.

pa_address unsigned int
: Physical address.

st2_st1_context_banks unsigned int
: The stage 1 and 2 context bank that this walk is for (if there is one, otherwise 0xFF). Stage 1 is the lowest byte, stage 2 the highest.

stage_and_level unsigned int
: Stage and level that this walk is for. Top byte is stage, bottom is level.

trans_id unsigned int
: Transaction id.

start_stall_transaction

Indicating that a transaction is stalled. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

context_id unsigned int
: The context id that we are stalling against.

head_of_line bool
: The transaction is the head-of-line to stall -- it is the one that will pick up the value of SMMU_CBn_RESUME.

reason_for_stall enum
: The state of the transaction as to why it stalled.

trans_id unsigned int
: Transaction id of replaying transaction.

tlb_commentary

A commentary from the TLB about what is going on. Fields:

output string
: The stream output.

tlb_entry_allocated

A TLB entry has been allocated. Fields:

index unsigned int
: Index of TLB entry.

input_end_incl_address unsigned int
: The end inclusive address of the input range that this matches.

input_start_address unsigned int
: The start address of the input range that this matches.

output_end_incl_address unsigned int
: The end inclusive address of the output range.

output_start_address unsigned int
: The start address of the output range.

tlb_entry_invalidated

A TLB entry has been invalidated for some reason. Fields:

index unsigned int
: Index of TLB entry.

reason enum
: The reason why it was invalidated.

tlb_invalidate_command_executed

A TLB Invalidate command was executed. Fields:

address unsigned int
: Address if appropriate.

asid unsigned int
: ASID if appropriate.

command enum
: Command queued

context_bank unsigned int
: Context bank number if queue_type is 'context_bank'.

last_level bool
: Last level if appropriate.

ns bool
: Security world if appropriate.

queue_type enum
: Queue added to

seq_id unsigned int
: Sequence id for this TLB invalidate.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

vmid unsigned int
VMID if appropriate.

tlb_invalidate_command_queued

A TLB Invalidate command was queued (but not synced). Fields:

address unsigned int
Address if appropriate.
asid unsigned int
ASID if appropriate.
command enum
Command queued
context_bank unsigned int
Context bank number if queue_type is 'context_bank'.
last_level bool
Last level if appropriate.
ns bool
Security world if appropriate.
queue_type enum
Queue added to
seq_id unsigned int
Sequence id for this TLB invalidate.
vmid unsigned int
VMID if appropriate.

tlb_sync

A TLB Sync command was issued. Fields:

context_bank unsigned int
Context bank number if queue_type is 'context_bank'.
ns bool
Security world
queue_type enum
Queues that will be synced

unpred_nested_st2_bank_is_not_marked_as_st2

The Stage 1 context bank is marked as Stage 1 + Stage 2 by SMMU_CBARn.TYPE; however, the Stage 2 context bank marked by SMMU_CBARn.CBNDX is not marked in it's SMMU_CBARm.TYPE as a Stage 2 context. Fields:

st1_context_bank_id unsigned int
The context bank id of the stage 1 bank.
st2_context_bank_id unsigned int
The context bank id of the stage 2 bank.

unpred_ptw_replay_asked_for_different_data

A PTW was in progress and was restarted (for some reason), but this time it asked for different data than last time. This means that the register file changed during the walk process and so the result of the translation for this transaction is unpredictable as to whether it would have seen the first or second walk data. Fields:

first_address unsigned int
The address of the first walk transaction that it performed.
first_ns bool
The NS state of the first walk transaction that it performed.
second_address unsigned int
The address that it is now walking for.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

second_ns bool
: The NS state that it is now walking for.

trans_id unsigned int
: Transaction ID that was affected.

v7sd_access_fault

A L1 or L2 short descriptor access flag has disabled access for this (super-)section/page. Fields:

context_bank unsigned int
: Context bank.

input_address unsigned int
: The input address used for this descriptor.

is_non_secure bool
: Is this for the Non-secure side?

level unsigned int
: Level of walk.

v7sd_descriptor_faults

V7 Short Descriptor, level 'n' descriptor is marked as faulting. Fields:

context_bank unsigned int
: Context bank.

input_address unsigned int
: The input address used for this descriptor.

is_non_secure bool
: Is this for the Non-secure side?

level unsigned int
: Level of walk.

v7sd_descriptor_fetch_failed

V7 Short Descriptor, level 'n' descriptor fetch failed. Fields:

context_bank unsigned int
: Context bank.

ipa_address unsigned int
: IPA of the descriptor address.

is_non_secure bool
: Is the page walk and input addresses non-secure?

level unsigned int
: Level of walk.

pa_address unsigned int
: PA of the descriptor address.

v7sd_walk_process_is_disabled

V7 Short Descriptor, walk process is disabled so producing a fault. Fields:

context_bank unsigned int
: Context bank.

warning

These messages are about unusual (but not necessarily incorrect) activity occurring on the SMMU. Fields:

output string
: The stream output.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

warning_failed_to_clear_fsr_before_resuming

The programming model expects for the user to clear the fault in the FSR before calling SMMU_CBn_RESUME to unstall a transaction. Fields:

context_id `unsigned int`
: Context ID of the FSR.

fault_flags_of_fsr `unsigned int`
: Value of the fault bits of the FSR (excludes SS and the Format field).

warning_multiple_match_transaction

A transaction was encountered that multiple matched the stream id registers. Fields:

action `enum`
: Action taken.

trans_id `unsigned int`
: Transaction id.

warning_reg_after_doesnt_match_written_value

A write occurred that tried to set bits in a register, that for one reason or another, failed to get written. Fields:

desc `string`
: The textual description of what happened.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.83 MMU_500_BASE - trace

This section describes the trace sources.

SMMU_SACR_change_PAGESIZE

The PAGESIZE of the device was changed by a write to the SMMU_SACR register. Fields:

is_64KiB bool
: The PAGESIZE is 64KiB.

aarch64_AddressSizeFault_base_address_out_of_range

An AArch64 context got an AddressSizeFault as the base address of the TTBR used is out of range. Fields:

CPAMax unsigned int
: Configured PAMax.

PAMax unsigned int
: Actual PAMax.

context_bank unsigned int
: context_bank.

stage unsigned int
: Stage.

aarch64_AddressSizeFault_in_final_block_address_trace

An AArch64 context got an AddressSizeFault in the final block descriptor. Fields:

CPAMax unsigned int
: Configured PAMax.

PAMax unsigned int
: Actual PAMax.

context_bank unsigned int
: context_bank.

output_address unsigned int
: Output address from stage.

stage unsigned int
: Stage.

aarch64_AddressSizeFault_in_table_descriptor

An AArch64 context got an AddressSizeFault as a descriptor encoded a page and the table base address described was out of range. Fields:

CPAMax unsigned int
: Configured PAMax.

PAMax unsigned int
: Actual PAMax.

context_bank unsigned int
: context_bank.

ipa_address unsigned int
: IPA of descriptor.

ns bool
: Descriptor address is non-secure.

pa_address unsigned int
: PA of descriptor.

stage unsigned int
: Stage.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

aarch64_TranslationFault_as_block_too_large

The AArch64 block described by a descriptor is too large. Fields:

context_bank unsigned int
: context_bank.

level unsigned int
: level.

stage unsigned int
: stage.

aarch64_TranslationFault_input_address_out_of_range

An AArch64 context received an address which was outside of the programmed TxSZ. Fields:

context_bank unsigned int
: context_bank.

input_address unsigned int
: Input address to stage.

ns bool
: Security state of context bank.

stage unsigned int
: Stage.

aarch64_TranslationFault_st2_starting_level_and_tsize_inconsistent

An AArch64 context got a TranslationFault as the specified starting level and TxSZ field are inconsistent. Fields:

context_bank unsigned int
: context_bank.

stage unsigned int
: Stage.

add_ns_context_fault

A context fault occurred and this is the record it wishes to generate. Fields:

SMMU_CBFRSYNRAn unsigned int
: SMMU_CBFRSYNRAn value that it wishes to write.

SMMU_CBn_FAR unsigned int
: SMMU_CBn_FAR value that it wishes to write.

SMMU_CBn_FSR unsigned int
: SMMU_CBn_FSR value that it wishes to write.

SMMU_CBn_FSYNR0 unsigned int
: SMMU_CBn_FSYNR0 value that it wishes to write.

SMMU_CBn_FSYNR1 unsigned int
: SMMU_CBn_FSYNR1 value that it wishes to write.

SMMU_CBn_IPAFAR unsigned int
: (SMMUv2 only) SMMU_CBn_IPAFAR value that it wishes to write. The value is UNKNOWN if this is a stage 1 context.

context_bank unsigned int
: Context bank.

going_to_be_lost bool
: This fault record is going to be lost as there is already an unhandled fault record.

add_ns_global_fault

Attempt to raise a Non-Secure global fault. Fields:

SMMU_GFAR unsigned int
: The fault address that we are attempting to record.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SMMU_GFSR unsigned int
: The bits that we are attempting to set in the SMMU_GFSR.

SMMU_GFSYNR0 unsigned int
: The syndrome we are attempting to record.

SMMU_GFSYNR1 unsigned int
: The syndrome we are attempting to record.

going_to_be_lost bool
: This fault record is going to be lost as there is already an unhandled fault record.

add_s_context_fault

A context fault occurred and this is the record it wishes to generate. Fields:

SMMU_CBFRSYNRAn unsigned int
: SMMU_CBFRSYNRAn value that it wishes to write.

SMMU_CBn_FAR unsigned int
: SMMU_CBn_FAR value that it wishes to write.

SMMU_CBn_FSR unsigned int
: SMMU_CBn_FSR value that it wishes to write.

SMMU_CBn_FSYNR0 unsigned int
: SMMU_CBn_FSYNR0 value that it wishes to write.

SMMU_CBn_FSYNR1 unsigned int
: SMMU_CBn_FSYNR1 value that it wishes to write.

SMMU_CBn_IPAFAR unsigned int
: (SMMUv2) SMMU_CBn_IPAFAR value that it wishes to read. This becomes UNKNOWN as this is a secure bank.

context_bank unsigned int
: Context bank.

going_to_be_lost bool
: This fault record is going to be lost as there is already an unhandled fault record.

add_s_global_fault

Attempt to raise a Secure global fault. Fields:

SMMU_SGFAR unsigned int
: The fault address that we are attempting to record.

SMMU_SGFSR unsigned int
: The bits that we are attempting to set in the SMMU_SGFSR.

SMMU_SGFSYNR0 unsigned int
: The syndrome we are attempting to record.

SMMU_SGFSYNR1 unsigned int
: The syndrome we are attempting to record.

going_to_be_lost bool
: This fault record is going to be lost as there is already an unhandled fault record.

ats_cb_atsr_read

A context bank Address Translation Status Register read occurred (Writes are reported through ats_cb_operation_doesnt_exist_trace). Fields:

ACTIVE bool
: The ACTIVE bit returned.

context_id unsigned int
: Context bank ID.

is_debug bool
: The access was a debug access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ats_cb_operation_doesnt_exist

An attempt was made to use a Context Bank Address Translation Operation that didn't exist for some reason. Fields:

context_id unsigned int
: Context bank ID.

desc string
: Description of why it doesn't exist.

ats_cb_operation_queued

A new Context Bank Address Translation Operation was queued. Fields:

VA unsigned int
: Virtual Address.

ats_is_privileged bool
: The ATS operation is asking for a 'privileged' access.

ats_is_read bool
: The ATS operation is asking for a 'read' access.

context_id unsigned int
: Context bank ID.

is_debug bool
: Is for a debug request.

ats_cb_operation_received_while_ats_in_progress

Another Context Bank Address Translation operation was received whilst one was currently in progress. This is UNPREDICTABLE. Fields:

context_id unsigned int
: Context bank ID.

desc string
: Description of new ATS operation.

ats_global_atsr_read

A Global Address Translation Status Register read occurred (Writes are reported through ats_cb_operation_doesnt_exist_trace). Fields:

ACTIVE bool
: The ACTIVE bit returned.

is_debug bool
: The access was a debug access.

which enum
: Which global ATSR register?

ats_global_operation_doesnt_exist

An attempt was made to use a Context Bank Address Translation Operation that didn't exist for some reason. Fields:

desc string
: Description of why it doesn't exist.

is_non_secure enum
: Is the operation non-secure.

ats_global_operation_queued

A new Context Bank Address Translation Operation was queued. Fields:

VA unsigned int
: Virtual Address.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ats_is_privileged bool
: The ATS operation is asking for a 'privileged' access.

ats_is_read bool
: The ATS operation is asking for a 'read' access.

context_id unsigned int
: Context bank ID.

is_debug bool
: Is for a debug request.

is_non_secure enum
: Is the operation non-secure.

is_s12 bool
: Is a stage 1 and stage 2 translation.

ats_global_operation_received_while_ats_in_progress

Another Global Address Translation operation was received whilst one was currently in progress. This is UNPREDICTABLE. Fields:

context_id unsigned int
: Context bank ID of new operation.

desc string
: Description of new ATS operation.

is_non_secure enum
: Is the operation non-secure.

ats_op_started

A queued address translation operation has started. Fields:

VA unsigned int
: Input virtual address.

context_bank unsigned int
: Context bank if which == 'context_bank'.

is_debug bool
: The operation is a debug ATS command.

is_privileged bool
: Is a privileged operation.

is_read bool
: Is a read operation.

is_s12 bool
: Is for a stage 1 and 2 translation.

which enum
: Which group of operations?

cfg_cttw

The cfg_cttw signal level that indicates that the SoC support coherent page walks. Fields:

value bool
: The signal level.

changed_PAGESIZE

The PAGESIZE has been told to change. Fields:

is_64KiB bool
: The PAGESIZE is 64 KiB.

changing_security_state_of_context_bank

The specified bank has changed its security state. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

context_bank unsigned int
: Context bank number.

now_is_non_secure bool
: The context bank is now non-secure.

prev_is_non_secure bool
: Prior to this change in the SMMU_SCR1.NSNUMCBO then the bank was non-secure.

cmo_permission_fault

A Cache Maintenance Operation (CMO) generated a permission fault. Fields:

abort_returned_to_device bool
: An abort is returned to the device.

cmo enum
: Cache Maintenance Operation.

context_id unsigned int
: Context ID that receives the fault.

input_address unsigned int
: The address of the CMO.

cmo_permission_model_being_applied

The cache maintenance operation permission model is being applied. Fields:

aarch64 bool
: Is the initial context bank aarch64?

fault_action enum
: Whether there is a fault or not.

incoming_cmo_operation enum
: The CMO operation.

instruction bool
: Is the CMO instruction?

outgoing_cmo_operation enum
: The CMO operation.

privileged bool
: Is the CMO privileged?

stage1_perms enum
: The permissions at stage 1.

stage2_perms enum
: The permissions at stage 1.

comb_irpt

Combined interrupt signal. This is the OR of all of the non-secure interrupt signals. Fields:

asserted bool
: The signal is asserted.

is_non_secure bool
: Is the comb_irpt_ns signal or the comb_irpt_s signal.

configuration_access_fault_unimplemented_stream_mapping_register_group

This occurs when a configuration transaction attempts to access a stream mapping register group that doesn't exist and if the implementation treats this as a Configuration Access Fault. Fields:

address unsigned int
: Address that was used.

group_number unsigned int
: Group number if it were to exist/be accessible.

is_non_secure bool
: Is the configuration access non-secure?

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

is_read bool
: Is the configuration access a read?

context_bank_commentary

These messages are verbose commentary about what the context bank is doing. Fields:

output string
: The stream output.

context_bank_mmu_off_stage1

This is the transformation applied by the stage 1 SMMU_CBn_SCTLR. Fields:

SMMU_CBn_SCTLR unsigned int
: Value of the SMMU_CBn_SCTLR used.

context_id unsigned int
: Context bank id the transaction has been mapped to.

trans_id unsigned int
: Transaction ID.

context_bank_mmu_off_stage2

This is the transformation applied by the stage 2 SMMU_CBn_SCTLR. Fields:

SMMU_CBn_SCTLR unsigned int
: Value of the SMMU_CBn_SCTLR used.

context_id unsigned int
: Context bank id the transaction has been mapped to.

trans_id unsigned int
: Transaction ID.

ctx_irpt

Context Interrupt. Fields:

asserted bool
: The signal is asserted.

context_id unsigned int
: The context id of the interrupt.

debug

This is the textual output of the debug messages of the SMMU model, this need not necessarily make much sense to the user, but is useful for sending to the model writers when reporting bugs. Fields:

output string
: The stream output.

denied_access_due_to_imp_def_reason

The register access was denied because of some IMPLEMENTATION DEFINED reason due to the transaction attributes. Fields:

address unsigned int
: Address of register access.

desc string
: Description of register.

denied_access_due_to_smmu_scr1_gasrae

The register access was denied because SMMU_SCR1.GASRAE has locked down the register. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

address unsigned int
: Address of register access.

dvm_tlbinvalidate_complete

The DVM TLB Invalidate message completed.

dvm_tlbinvalidate_received

A DVM message for a TLB Invalidate has been received. Fields:

address unsigned int
: The VA or IPA to use if match_address.

asid unsigned int
: The ASID to match if match_asid.

by_ipa bool
: The operation is for an IPA operation if match_address.

ignored enum
: The DVM message was ignored.

last_level bool
: The operation is for last level if supported.

match_address bool
: Match the address field.

match_asid bool
: Match the asid field.

match_vmid bool
: Match the vmid field.

prot enum
: The protection level for which this TLB Invalidate will operate on.

security_world enum
: The security world that this will apply to.

stage1_only bool
: The operation is for stage 1 only if supported.

vmid unsigned int
: The VMID to match if match_vmid.

end_stall_transaction

Indicates a transaction unstalling. Fields:

context_id unsigned int
: The context id that we are stalling against.

trans_id unsigned int
: Transaction id of replaying transaction.

unstall_state enum
: Describes why we unstalled. If we are head of line then we are either replay/terminate, otherwise we will be not be.

error

These messages are about activity occurring on the SMMU that is considered an error. Fields:

output string
: The stream output.

error_contiguous_bit_is_bad

The contiguous bit is inconsistent with the descriptor. Fields:

B_mask unsigned int
: The mask of the address that is zapped if we were to use the contiguous bit.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

aarch64_contig_bit_too_high bool
: AArch64 claims that the contig bit can't be set for this particular set of start levels and input address size.

input_address unsigned int
: The input address that will be combined with the output address.

ln2_size_in_bytes_of_region_if_contiguous unsigned int
: The number of address bits that form the 'offset' within the contiguous region.

out_of_range_of_input bool
: The range mapped by a contiguous bit being on is larger than the input range for this stage.

output_address unsigned int
: The output address as contained in the descriptor.

output_address_if_contig_bit_uses_input_address unsigned int
: Output address if contig bit was fully taken (maximally sized region) and those bits came from the input address.

output_address_inconsistent bool
: The output address in the descriptor is inconsistent with its position in the table.

stage_and_level unsigned int
: Top 8 bits are the stage, bottom 8 bits are level.

error_invalid_smmu_cbar

A SMMU_CBAR has been encountered for a secure stream that is not of the Stage 1 with Stage 2 bypass form. Fields:

SMMU_CBARn unsigned int
: The corresponding SMMU_CBARn for the context.

action enum
: Action taken.

context_id unsigned int
: Context ID.

trans_id unsigned int
: Transaction id.

error_tlb_entries_overlap

A TLB entry was attempted to be inserted into the TLB and was determined that it overlaps an existing entry. This check is not perfect but will catch simple errors. Fields:

end_address_of_new_entry unsigned int
: End address of new entry.

end_address_of_old_entry unsigned int
: End address of old entry.

index_of_new_entry unsigned int
: Index of new entry.

index_of_old_entry unsigned int
: Index of old entry.

start_address_of_new_entry unsigned int
: Start address of new entry.

start_address_of_old_entry unsigned int
: Start address of old entry.

extended_attributes_for_stage

This gives the attributes that a particular stage gives prior to being applied to the incoming attributes. Fields:

inner enum
: Inner cacheability.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ns bool
: NS/S security state.

outer enum
: Outer cacheability.

racfg enum
: RACFG from descriptor (if exists).

shareability enum
: Shareability.

stage unsigned int
: Stage that this record is for.

trans_id unsigned int
: Transaction ID this is for.

wacfg enum
: WACFG from descriptor (if exists).

glbl_flt_irpt

Global Fault Interrupt. Fields:

asserted bool
: The signal is asserted.

is_non_secure bool
: glbl_flt_irpt_ns or glbl_flt_irpt_s.

imp_def_non_SMMU_CBn_RESUME_register_access_transaction_forces_replay

There is an IMP DEF option to replay stalled transactions in affected banks. This records that the implementation has made use of this option. Fields:

context_id unsigned int
: Context ID of affected bank.

why enum
: Why did this replay occur?

implementation_postprocess

This describes the non-architectural transformation that the implementation makes. Fields:

end_adomain enum
: End adomain value.

end_inner_acache enum
: End inner acache value.

end_outer_acache enum
: End outer acache value.

is_read bool
: Is a read transaction.

start_adomain enum
: Start adomain value.

start_inner_acache enum
: Start inner acache value.

start_outer_acache enum
: Start outer acache value.

info

These are information messages about what is happening in the SMMU. Fields:

output string
: The stream output.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

invalid_address_range

An input or output address range check failed. Input address range checks are reported as TranslationFault, output ones as AddressSizeFault. Fields:

context_bank unsigned int
: Context bank.

offending_address unsigned int
: The address that failed the address check.

why enum
: The reason why the address was invalid.

ld_access_fault

An Access Fault occurred. Fields:

context_bank unsigned int
: Context bank.

input_address unsigned int
: The input address used for this descriptor.

is_non_secure bool
: Is the input address non-secure?

level unsigned int
: Level of walk.

stage unsigned int
: Stage of walk.

ld_descriptor_fetch_failed

Long Descriptor AArch32 or AArch64, level 'n' descriptor fetch failed. Fields:

context_bank unsigned int
: Context bank.

ipa_address unsigned int
: IPA of the descriptor address.

is_non_secure bool
: Is the page walk and input addresses non-secure?

level unsigned int
: Level of walk.

pa_address unsigned int
: PA of the descriptor address.

stage unsigned int
: Stage of walk.

ld_descriptor_invalid

Long Descriptor AArch32 or AArch64 encodes an invalid entry. Fields:

context_bank unsigned int
: Context bank.

ipa_address unsigned int
: IPA of the descriptor address.

level unsigned int
: Level of walk.

pa_address unsigned int
: PA of the descriptor address.

ssd_ns bool
: Is the transaction classified as non-secure.

stage unsigned int
: Stage of walk.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ld_hyp_mode_address_out_of_range

A hypervisor input address is out of range of T0SZ. Fields:

context_bank unsigned int
: The context bank.

input_address unsigned int
: The VA input address used for this descriptor.

is_non_secure bool
: Is this for the Non-secure side?

ld_ipa_out_of_range

A stage 2 context bank was given an address that didn't match T0SZ. Fields:

context_bank unsigned int
: The context bank.

input_address_ipa unsigned int
: The IPA input address used for this descriptor.

is_non_secure bool
: Is this for the Non-secure side?

ld_ipa_out_of_range_for_st1ptw_st2_mmu_off

(SMMUv2) The stage 2 context (nested with MMU off or bypass) for a stage 1 translation had an address that was out of the allowed range. Fields:

context_bank unsigned int
: The context bank.

input_address_ipa unsigned int
: The IPA input address used for this descriptor.

is_non_secure bool
: Is this for the Non-secure side?

ld_last_level_descriptor_does_not_encode_a_page

Long Descriptor, last level descriptor must encode a page but didn't. Fields:

context_bank unsigned int
: Context bank.

ipa_address unsigned int
: IPA of the descriptor address.

level unsigned int
: Level of walk.

pa_address unsigned int
: PA of the descriptor address.

ssd_ns bool
: Is the transaction classified as non-secure?

stage unsigned int
: Stage of walk.

ld_misprogramming_fault_trace

A misprogramming occurred that the implementation faulted. Fields:

context_bank unsigned int
: Context bank.

input_address unsigned int
: Input address.

stage unsigned int
: Stage.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

why enum
: Why the fault occurred.

ld_st2_produced_any_device_memory_for_st1ptw

A stage 2 translation process for a stage 1 page walk address produce a any-device memory type. Fields:

context_bank unsigned int
: Context bank.

input_address unsigned int
: The input address used for this descriptor.

level unsigned int
: Level of walk.

ld_st2_s1ptw_has_no_access_rights

A stage 2 translation process for a stage 1 page walk address produce a permission fault. Fields:

context_bank unsigned int
: Context bank.

input_address unsigned int
: The input address used for this descriptor.

level unsigned int
: Level of walk.

ld_stage1_non_hyp_no_ttbr_match

A stage 1 non-hyp entry failed to match TTBR0 or TTBR1. Fields:

context_bank unsigned int
: The context bank.

input_address unsigned int
: The input address used.

is_non_secure bool
: Is this for the Non-secure side?

ld_walk_process_is_disabled

A L1 or L2 short descriptor access flag has disabled access for this (super-)section/page. Fields:

context_bank unsigned int
: The context bank.

input_address unsigned int
: The input address used.

is_non_secure bool
: Is this for the Non-secure side?

stage unsigned int
: Which change this corresponds to.

ttbr_id unsigned int
: Which TTBR we are using (is 0 for a stage 2).

non_PTW_external_fault_recorded

A non Page Table Walk access was recorded when a translated transaction went downstream, but the downstream system aborted it. This is recorded as an asynchronous fault. In the model, we can provide more information about the faulting transaction. Fields:

downstream_address unsigned int
: The downstream address.

downstream_ns enum
: The downstream NS state. This is not the SSD.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

is_read enum
: Is the transaction a read?

result_reported_upstream enum
: The result that is reported upstream.

sGFSYNR0 unsigned int
: The value of sGFSYNR0 that will attempt to be recorded (all 1s mean it will not be updated).

sGFSYNR1 unsigned int
: The value of sGFSYNR1 that will attempt to be recorded (all 1s mean it will not be updated).

smmu_scr1_gefro bool
: The value of SMMU_SCR1.GEFRO.

smmu_xcr0_gfre bool
: The appropriate SMMU_sCR0.GFRE bit. If this is 0 then it squash the downstream abort.

ssd_ns enum
: SSD.

upstream_address unsigned int
: The upstream address.

upstream_ns enum
: The upstream NS state. This is not the SSD.

post_s2crn_attributes

The attributes after the SMMU_S2CRn transformation has been applied. Fields:

adomain enum
: The actual attributes of the access that was used, after IMP DEF mangling.

aprot enum
: The actual attributes of the access that was used, after IMP DEF mangling.

inner_cache enum
: The actual attributes of the access that was used, after IMP DEF mangling.

inner_transient bool
: Is the transaction marked with the transient hint?

outer_cache enum
: The actual attributes of the access that was used, after IMP DEF mangling.

outer_transient bool
: Is the transaction marked with the transient hint?

trans_id unsigned int
: Transaction id.

v8_dev_gathering bool
: Is the transaction marked with the V8 Device Gathering property?

v8_dev_reordering bool
: Is the transaction marked with the V8 Device Reordering property?

ptw_read

Page Table Walk (read). This is the physical access that the SMMU is making. Fields:

adomain enum
: The actual attributes of the access that was used, after IMP DEF mangling.

aprot enum
: The actual attributes of the access that was used, after IMP DEF mangling.

auser unsigned int
: The AUSER fields.

data unsigned int
: The data read in little-endian format.

hyp bool
: Current mode is hyp.

inner_cache enum
: The actual attributes of the access that was used, after IMP DEF mangling.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

input_address unsigned int
: Input address this lookup is for.

ns bool
: This is for SSD Non-secure.

outer_cache enum
: The actual attributes of the access that was used, after IMP DEF mangling.

pa_address unsigned int
: Physical address.

result enum
: The PTW result.

st2_st1_context_banks unsigned int
: The stage 1 and 2 context bank that this walk is for (if there is one, otherwise 0xFF). Stage 1 is the lowest byte, stage 2 the highest.

stage_and_level unsigned int
: Stage and level that this walk is for. Top byte is stage, bottom is level.

trans_id unsigned int
: Transaction id.

ptw_read_st1_invalid_long_descriptor

Page Table Walk (read). The descriptor that the PTW fetched is invalid. Fields:

hyp bool
: Current mode is hyp.

input_address unsigned int
: Input address this lookup is for.

level unsigned int
: The page table level this walk is for.

ns bool
: This is for SSD Non-secure.

pa_address unsigned int
: Physical address.

st1_context_bank unsigned int
: The stage 1 context bank that this walk is for (if there is one, otherwise 0xFF).

st2_context_bank unsigned int
: The stage 2 context bank that this walk is for (if there is one, otherwise 0xFF).

trans_id unsigned int
: Transaction id.

ptw_read_st1_leaf_long_descriptor

Page Table Walk (read). The descriptor that the PTW fetched is a block or page and this is the decode. Fields:

AF bool
: Access Flag.

AP21 enum
: The access permissions.

AttrIndx210 unsigned int
: The attribute index into the MAIR0/1.

NS bool
: The encoding is for non-secure if this is a secure fetch.

PXN bool
: Privileged eXecute Never.

SH10 enum
: The shareability.

XN bool
: eXecute Never.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

contiguous bool
: Contiguous hint.

hyp bool
: Current mode is hyp.

input_address unsigned int
: Input address this lookup is for.

level unsigned int
: The page table level this walk is for.

mair unsigned int
: The MAIRn encoding attribute.

nG bool
: not Global.

ns bool
: The agent is Non-Secure.

output_address unsigned int
: Output address.

pa_address unsigned int
: Physical address.

st1_context_bank unsigned int
: The stage 1 context bank that this walk is for (if there is one, otherwise 0xFF).

st2_context_bank unsigned int
: The stage 2 context bank that this walk is for (if there is one, otherwise 0xFF).

trans_id unsigned int
: Transaction id.

ptw_read_st1_table_long_descriptor

Page Table Walk (read). The descriptor that the PTW fetched is a Table, this decodes the fields. Fields:

APTable enum
: Remove permissions independently of subsequent descriptors.

NSTable bool
: The next level table descriptor is forced to non-secure.

PXNTable bool
: Force PXN independently of subsequent descriptors.

TableAddress unsigned int
: Address of the next table.

XNTable bool
: Force XN independently of subsequent descriptors.

hyp bool
: Current mode is hyp.

input_address unsigned int
: Input address this lookup is for.

level unsigned int
: The page table level this walk is for.

ns bool
: This is for SSD Non-secure.

pa_address unsigned int
: Physical address.

st1_context_bank unsigned int
: The stage 1 context bank that this walk is for (if there is one, otherwise 0xFF).

st2_context_bank unsigned int
: The stage 2 context bank that this walk is for (if there is one, otherwise 0xFF).

trans_id unsigned int
: Transaction id.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ptw_read_st1_v7s_fault_descriptor

Page Table Walk (read). The descriptor encodes a fault descriptor. Fields:

level unsigned int
: The level of the page table.

trans_id unsigned int
: Transaction id.

ptw_read_st1_v7s_large_page_descriptor

Page Table Walk (read). The descriptor encodes a large page descriptor. Fields:

AP210 unsigned int
: AP[2:0].

S bool
: Shareable.

TEX210CB unsigned int
: {TEX[2:0],C,B}.

XN bool
: eXecute Never.

large_page_base_address unsigned int
: Base address of the large page.

nG bool
: not Global.

trans_id unsigned int
: Transaction id.

ptw_read_st1_v7s_page_table_descriptor

Page Table Walk (read). The descriptor encodes a page table descriptor pointing to an L2 table. Fields:

NS bool
: Encodes a non-secure entry.

PXN bool
: Privileged eXecute Never.

domain unsigned int
: Domain.

l2_table_address unsigned int
: L2 page table address.

trans_id unsigned int
: Transaction id.

ptw_read_st1_v7s_section_descriptor

Page Table Walk (read). The descriptor encodes a section descriptor. Fields:

AP210 unsigned int
: AP[2:0].

NS bool
: Encodes a non-secure entry.

PXN bool
: Privileged eXecute Never.

S bool
: Shareable.

TEX210CB unsigned int
: {TEX[2:0],C,B}.

XN bool
: eXecute Never.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

domain unsigned int
: Domain.

nG bool
: not Global.

section_base_address unsigned int
: Base address of section.

trans_id unsigned int
: Transaction id.

ptw_read_st1_v7s_small_page_descriptor

Page Table Walk (read). The descriptor encodes a small page descriptor. Fields:

AP210 unsigned int
: AP[2:0].

S bool
: Shareable.

TEX210CB unsigned int
: {TEX[2:0],C,B}.

XN bool
: eXecute Never.

nG bool
: not Global.

small_page_base_address unsigned int
: Base address of the small page.

trans_id unsigned int
: Transaction id.

ptw_read_st1_v7s_super_section_descriptor

Page Table Walk (read). The descriptor encodes a section descriptor. Fields:

AP210 unsigned int
: AP[2:0].

NS bool
: Encodes a non-secure entry.

PXN bool
: Privileged eXecute Never.

S bool
: Shareable.

TEX210CB unsigned int
: {TEX[2:0],C,B}.

XN bool
: eXecute Never.

nG bool
: not Global.

super_section_base_address unsigned int
: Base address of the super section.

trans_id unsigned int
: Transaction id.

ptw_read_st2_invalid_descriptor

Page Table Walk (read). The descriptor that the PTW fetched is invalid. Fields:

hyp bool
: Current mode is hyp.

input_address unsigned int
: Input address this lookup is for.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

level unsigned int
: The page table level this walk is for.

ns bool
: This is for SSD Non-secure.

pa_address unsigned int
: Physical address.

st1_context_bank unsigned int
: The stage 1 context bank that this walk is for (if there is one, otherwise 0xFF).

st2_context_bank unsigned int
: The stage 2 context bank that this walk is for (if there is one, otherwise 0xFF).

trans_id unsigned int
: Transaction id.

ptw_read_st2_leaf_descriptor

Page Table Walk (read). The descriptor that the PTW fetched is a block or page and this is the decode. Fields:

AF bool
: Access Flag.

HAP21 enum
: The access permissions.

MemAttr3_0 enum
: The memory attributes.

RACFG enum
: The extra SMMU RACFG field.

SH10 enum
: The shareability.

WACFG enum
: The extra SMMU WACFG field.

XN bool
: eXecute Never.

contiguous bool
: Contiguous hint.

hyp bool
: Current mode is hyp.

input_address unsigned int
: Input address this lookup is for.

level unsigned int
: The page table level this walk is for.

ns bool
: This is for SSD Non-secure.

output_address unsigned int
: Output address.

pa_address unsigned int
: Physical address.

st1_context_bank unsigned int
: The stage 1 context bank that this walk is for (if there is one, otherwise 0xFF).

st2_context_bank unsigned int
: The stage 2 context bank that this walk is for (if there is one, otherwise 0xFF).

trans_id unsigned int
: Transaction id.

ptw_read_st2_table_descriptor

Page Table Walk (read). The descriptor that the PTW fetched is a Table, this decodes the fields. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

APTable enum
: Remove permissions independently of subsequent descriptors.

NSTable bool
: The next level table descriptor is forced to non-secure.

PXNTable bool
: Force PXN independently of subsequent descriptors.

TableAddress unsigned int
: Address of the next table.

XNTable bool
: Force XN independently of subsequent descriptors.

hyp bool
: Current mode is hyp.

input_address unsigned int
: Input address this lookup is for.

level unsigned int
: The page table level this walk is for.

ns bool
: This is for SSD Non-secure.

pa_address unsigned int
: Physical address.

st1_context_bank unsigned int
: The stage 1 context bank that this walk is for (if there is one, otherwise 0xFF).

st2_context_bank unsigned int
: The stage 2 context bank that this walk is for (if there is one, otherwise 0xFF).

trans_id unsigned int
: Transaction id.

raw_want_to_change_context_interrupt

This represents a context's bank intent to want to change an interrupt pin. However, this doesn't take into account the accessibility of the bank, nor if other banks are driving the pin. Fields:

accessible bool
: Is the interrupt ID accessible to this context bank?

asserted bool
: Is this context bank attempting to assert the interrupt?

context_bank unsigned int
: Context bank attempting to change the interrupt pin.

interrupt_id unsigned int
: The interrupt ID it is attempting to change.

register_disallowed_read_string

A text representation of the read of a register that was disallowed. Fields:

out string
: The text description of the register value read

register_disallowed_write_string

A text representation of the write of a register write that was disallowed. Fields:

in string
: The text description of the register value written

register_read

Read of the register file. Fields:

address unsigned int
: The address of the register.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

context_id unsigned int
: Context id bank (or 0xFF if not a context bank)

ln2_width enum
: The log2 width of the access

ns enum
: The security state of the transaction addressing the register

register_trans_id unsigned int
: Register transaction ID (if top bit set then a debug transaction)

result enum
: The result of the access

value unsigned int
: The byte-stream of all the data (little-endian)

register_read_reserved

A text representation of an access to a register address that is reserved. Fields:

in string
: The text description of the register value

register_read_string

A text representation of the read of a register. Fields:

out string
: The text description of the register value read

register_write

Write to the register file. Fields:

address unsigned int
: The address of the register.

context_id unsigned int
: Context id bank (or 0xFF if not a context bank)

ln2_width enum
: The log2 width of the access

ns enum
: The security state of the transaction addressing the register

register_trans_id unsigned int
: Register transaction ID (if top bit set then a debug transaction)

result enum
: The result of the access

value unsigned int
: The byte-stream of all the data (little-endian)

register_write_reserved

A text representation of an access to a register address that is reserved. Fields:

in string
: The text description of the register value

register_write_string

A text representation of the write of a register. Fields:

in string
: The text description of the register value written

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

rejected_register_transaction

The transaction is rejected because it didn't have the right characteristics. Fields:

address unsigned int
: Address of transaction.

is_fixed bool
: Is a FIXED AMBA burst.

is_read bool
: Is a read.

number_of_beats unsigned int
: Number of beats in transaction.

width_in_bytes unsigned int
: Width of transaction in bytes.

reset

The reset signal level. Fields:

value bool
: The signal level.

reset_sequence

Marks the beginning and end of the reset sequence. Fields:

starting bool
: True if starting the reset sequence, false when finished.

smmu_faulting_transaction

This is a trace of transaction that are not expected to fault. This does not include those transaction that are faulting by stage 2 or where the access flag or permission model is not allowing the transaction. Fields:

context_id unsigned int
: The context id of the transaction used, if we failed to immediately hit in the TLB.

input_address unsigned int
: The input address of the transaction group.

mapped_attributes unsigned int
: The mapped attributes of the transaction.

original_attributes unsigned int
: The original unmapped attributes of the transaction.

output_address unsigned int
: The output address of the transaction group.

ssd_index unsigned int
: The Security State Determination Index if applicable.

ssd_ns enum
: The Security State Determination of the transaction.

state enum
: The state of the transaction.

stream_id unsigned int
: The stream id of the transaction

tbu unsigned int
: Translation Buffer Unit number.

tlb_entry_index_st1 unsigned int
: If this was mapped by a TLB entry index then this is the one used for the stage 1 translation.

tlb_entry_index_st2 unsigned int
: If this was mapped by a TLB entry index then this is the one used for the stage 2 translation.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

trans_id unsigned int
: Transaction id.

type_of_transaction enum
: The type of transaction.

smmu_final_transaction

This is a trace of the SMMU transaction after it has been fully remapped. Fields:

context_id unsigned int
: The context id of the transaction used, if we failed to immediately hit in the TLB.

input_address unsigned int
: The input address of the transaction group.

mapped_attributes unsigned int
: The mapped attributes of the transaction.

original_attributes unsigned int
: The original unmapped attributes of the transaction.

output_address unsigned int
: The output address of the transaction group.

ssd_index unsigned int
: The Security State Determination Index if applicable.

ssd_ns enum
: The Security State Determination of the transaction.

state enum
: The state of the transaction.

stream_id unsigned int
: The stream id of the transaction

tbu unsigned int
: Translation Buffer Unit number.

tlb_entry_index_st1 unsigned int
: If this was mapped by a TLB entry index then this is the one used for the stage 1 translation.

tlb_entry_index_st2 unsigned int
: If this was mapped by a TLB entry index then this is the one used for the stage 2 translation.

trans_id unsigned int
: Transaction id.

type_of_transaction enum
: The type of transaction.

smmu_initial_transaction

This is the initial transaction group request to remap. This represents a bundle of transactions with the same attributes but different addresses within a certain range around the address. Fields:

input_address unsigned int
: The input address of the transaction group.

kind_addressed enum
: The kind of transaction this represents if it is an addressed transaction

original_attributes unsigned int
: The original unmapped attributes of the transaction.

ssd_index unsigned int
: The Security State Determination Index if applicable.

ssd_ns enum
: The Security State Determination of the transaction.

stream_id unsigned int
: The stream id of the transaction.

tbu unsigned int
: Translation Buffer Unit number.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

trans_id unsigned int
: Transaction id.

type_of_transaction enum
: The type of transaction.

smmu_iprt_xgirpt

This trace source records the state of the SMMU_gIrpt/SMMU_NSgIrpt pins. Fields:

asserted bool
: The interrupt is now asserted.

is_non_secure bool
: Identifies the pin as SMMU_NSgIrpt.

smmu_irpt_combined_context_interrupt_pin

This trace source records when there is at least one context bank in the specific world with an interrupt pin raised. Fields:

asserted bool
: The interrupt is now asserted.

ns bool
: Non-secure

smmu_irpt_context_interrupt_pin

This trace source records when a context bank interrupt pin changes state. Fields:

asserted bool
: The interrupt is now asserted.

interrupt_pin unsigned int
: Interrupt pin number.

smmu_irpt_context_interrupt_pin_drivers

This trace source records when a context bank tries to assert an interrupt. Multiple banks can drive the same pin and so long as one is driving it then it is asserted. Fields:

context_bank unsigned int
: The context bank number changing its request for the pin

interrupt_pin unsigned int
: Interrupt pin number.

new_count_asserting_pin unsigned int
: The number of banks asserting the pin

this_context_asserting bool
: The request that this context is making -- if this is true then the context bank is asserting a request

smmu_irpt_xgcfgirpt

This trace source records the state of the SMMU_gCfgIrpt/SMMU_NSgCfgIrpt pins. Fields:

asserted bool
: The interrupt is now asserted.

is_non_secure bool
: Identifies the pin as SMMU_NSgCfgIrpt.

smmu_st1_st2fault_transaction

This is a trace of transactions that map to stage 1 with stage 2 faulting. Fields:

context_id unsigned int
: The context id of the transaction used, if we failed to immediately hit in the TLB.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

input_address unsigned int
: The input address of the transaction group.

mapped_attributes unsigned int
: The mapped attributes of the transaction.

original_attributes unsigned int
: The original unmapped attributes of the transaction.

output_address unsigned int
: The output address of the transaction group.

ssd_index unsigned int
: The Security State Determination Index if applicable.

ssd_ns enum
: The Security State Determination of the transaction.

state enum
: The state of the transaction.

stream_id unsigned int
: The stream id of the transaction

tbu unsigned int
: Translation Buffer Unit number.

tlb_entry_index_st1 unsigned int
: If this was mapped by a TLB entry index then this is the one used for the stage 1 translation.

tlb_entry_index_st2 unsigned int
: If this was mapped by a TLB entry index then this is the one used for the stage 2 translation.

trans_id unsigned int
: Transaction id.

type_of_transaction enum
: The type of transaction.

smrg_matched

Stream Match Register Group matched. Fields:

desc string
: A textual description of the register group

n unsigned int
: The SMRG that we matched against.

trans_id unsigned int
: The transaction id that we are matching against.

start_ptw_read

Page Table Walk (read). This is the start of the physical access that the SMMU is making. Fields:

adomain enum
: The actual attributes of the access that was used, after IMP DEF mangling.

aprot enum
: The actual attributes of the access that was used, after IMP DEF mangling.

auser unsigned int
: The AUSER fields.

hyp bool
: Current mode is hyp.

inner_cache enum
: The actual attributes of the access that was used, after IMP DEF mangling.

input_address unsigned int
: Input address this lookup is for.

ns bool
: This is for SSD Non-secure.

outer_cache enum
: The actual attributes of the access that was used, after IMP DEF mangling.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

pa_address unsigned int
: Physical address.

st2_st1_context_banks unsigned int
: The stage 1 and 2 context bank that this walk is for (if there is one, otherwise 0xFF). Stage 1 is the lowest byte, stage 2 the highest.

stage_and_level unsigned int
: Stage and level that this walk is for. Top byte is stage, bottom is level.

trans_id unsigned int
: Transaction id.

start_stall_transaction

Indicating that a transaction is stalled. Fields:

context_id unsigned int
: The context id that we are stalling against.

head_of_line bool
: The transaction is the head-of-line to stall -- it is the one that will pick up the value of SMMU_CBn_RESUME.

reason_for_stall enum
: The state of the transaction as to why it stalled.

trans_id unsigned int
: Transaction id of replaying transaction.

tlb_commentary

A commentary from the TLB about what is going on. Fields:

output string
: The stream output.

tlb_entry_allocated

A TLB entry has been allocated. Fields:

index unsigned int
: Index of TLB entry.

input_end_incl_address unsigned int
: The end inclusive address of the input range that this matches.

input_start_address unsigned int
: The start address of the input range that this matches.

output_end_incl_address unsigned int
: The end inclusive address of the output range.

output_start_address unsigned int
: The start address of the output range.

tlb_entry_invalidated

A TLB entry has been invalidated for some reason. Fields:

index unsigned int
: Index of TLB entry.

reason enum
: The reason why it was invalidated.

tlb_invalidate_command_executed

A TLB Invalidate command was executed. Fields:

address unsigned int
: Address if appropriate.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

asid unsigned int
: ASID if appropriate.

command enum
: Command queued

context_bank unsigned int
: Context bank number if queue_type is 'context_bank'.

last_level bool
: Last level if appropriate.

ns bool
: Security world if appropriate.

queue_type enum
: Queue added to

seq_id unsigned int
: Sequence id for this TLB invalidate.

vmid unsigned int
: VMID if appropriate.

tlb_invalidate_command_queued

A TLB Invalidate command was queued (but not synced). Fields:

address unsigned int
: Address if appropriate.

asid unsigned int
: ASID if appropriate.

command enum
: Command queued

context_bank unsigned int
: Context bank number if queue_type is 'context_bank'.

last_level bool
: Last level if appropriate.

ns bool
: Security world if appropriate.

queue_type enum
: Queue added to

seq_id unsigned int
: Sequence id for this TLB invalidate.

vmid unsigned int
: VMID if appropriate.

tlb_sync

A TLB Sync command was issued. Fields:

context_bank unsigned int
: Context bank number if queue_type is 'context_bank'.

ns bool
: Security world

queue_type enum
: Queues that will be synced

unpred_nested_st2_bank_is_not_marked_as_st2

The Stage 1 context bank is marked as Stage 1 + Stage 2 by SMMU_CBARn.TYPE; however, the Stage 2 context bank marked by SMMU_CBARn.CBNDX is not marked in it's SMMU_CBARm.TYPE as a Stage 2 context. Fields:

st1_context_bank_id unsigned int
: The context bank id of the stage 1 bank.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

st2_context_bank_id unsigned int
The context bank id of the stage 2 bank.

unpred_ptw_replay_asked_for_different_data

A PTW was in progress and was restarted (for some reason), but this time it asked for different data than last time. This means that the register file changed during the walk process and so the result of the translation for this transaction is unpredictable as to whether it would have seen the first or second walk data. Fields:

first_address unsigned int
The address of the first walk transaction that it performed.
first_ns bool
The NS state of the first walk transaction that it performed.
second_address unsigned int
The address that it is now walking for.
second_ns bool
The NS state that it is now walking for.
trans_id unsigned int
Transaction ID that was affected.

v7sd_access_fault

A L1 or L2 short descriptor access flag has disabled access for this (super-)section/page. Fields:

context_bank unsigned int
Context bank.
input_address unsigned int
The input address used for this descriptor.
is_non_secure bool
Is this for the Non-secure side?
level unsigned int
Level of walk.

v7sd_descriptor_faults

V7 Short Descriptor, level 'n' descriptor is marked as faulting. Fields:

context_bank unsigned int
Context bank.
input_address unsigned int
The input address used for this descriptor.
is_non_secure bool
Is this for the Non-secure side?
level unsigned int
Level of walk.

v7sd_descriptor_fetch_failed

V7 Short Descriptor, level 'n' descriptor fetch failed. Fields:

context_bank unsigned int
Context bank.
ipa_address unsigned int
IPA of the descriptor address.
is_non_secure bool
Is the page walk and input addresses non-secure?
level unsigned int
Level of walk.
pa_address unsigned int
PA of the descriptor address.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

v7sd_walk_process_is_disabled

V7 Short Descriptor, walk process is disabled so producing a fault. Fields:

context_bank unsigned int
: Context bank.

warning

These messages are about unusual (but not necessarily incorrect) activity occurring on the SMMU. Fields:

output string
: The stream output.

warning_failed_to_clear_fsr_before_resuming

The programming model expects for the user to clear the fault in the FSR before calling SMMU_CBn_RESUME to unstall a transaction. Fields:

context_id unsigned int
: Context ID of the FSR.

fault_flags_of_fsr unsigned int
: Value of the fault bits of the FSR (excludes SS and the Format field).

warning_multiple_match_transaction

A transaction was encountered that multiple matched the stream id registers. Fields:

action enum
: Action taken.

trans_id unsigned int
: Transaction id.

warning_reg_after_doesnt_match_written_value

A write occurred that tried to set bits in a register, that for one reason or another, failed to get written. Fields:

desc string
: The textual description of what happened.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.84 MMU_600 - trace

This section describes the trace sources.

ArchMsg.Error.error

These messages are about activity occurring on the SMMU that is considered an error. Messages will only come out here if parameter all_error_messages_through_trace is true. Fields:

output string
: The stream output

ArchMsg.Error.fetch_from_memory_type_not_supporting_httu

A descriptor fetch from an HTTU-enabled translation regime to an unsupported memory type was made. Whilst the fetch itself may succeed, if an update to the descriptor was attempted then it would fail. Fields:

address unsigned int
: The address of the descriptor fetch.

desc_inner enum
: Inner cacheability of descriptor

desc_outer enum
: Outer cacheability of descriptor

desc_prot enum
: The instruction/data, bus-ns/s and privileged/user marking

desc_sh enum
: Shareability of descriptor

ssd_ns bool
: Is the Security State Determination of the transaction non-secure.

stage unsigned int
: The stage at which we had a problem.

streamid unsigned int
: The StreamID of the transaction

substreamid unsigned int
: The SubstreamID of the transaction (or ~0u if no substreamid)

trans_id unsigned int
: Transaction id.

ArchMsg.Error.ns_cmdq

There is an error associated with the command queue. Fields:

cmd_id unsigned int
: Command ID of the command in error.

cons unsigned int
: CONS of the command.

what string
: What is wrong.

ArchMsg.Error.priq_streamid_truncated

The SMMU received a PCIe PRI request with a StreamID) that was larger than that which the SMMU has been configured for. The StreamID that appears in the PRIQ entry will be truncated. Fields:

actual_streamid unsigned int
: The actual StreamID that this request has.

sidsize unsigned int
: The bit width of the SMMU for StreamIDs, as indicated by SMMU_IDR1.SIDSIZE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

trans_id unsigned int
The transaction ID that identifies this request.
trunc_streamid unsigned int
The truncated StreamID that will appear in the PRIQ entry.

ArchMsg.Error.priq_substreamid_truncated

The SMMU received a PCIe PRI request with a PASID prefix (SubstreamID) that was larger than that which the SMMU has been configured for. The SubstreamID that appears in the PRIQ entry will be truncated. Fields:

actual_substreamid unsigned int
The actual SubstreamID that this request has.
ssidsize unsigned int
The bit width of the SMMU for SubstreamIDs, as indicated by SMMU_IDR1.SSIDSIZE.
trans_id unsigned int
The transaction ID that identifies this request.
trunc_substreamid unsigned int
The truncated SubstreamID that will appear in the PRIQ entry.

ArchMsg.Error.s_cmdq

There is an error associated with the command queue. Fields:

cmd_id unsigned int
Command ID of the command in error.
cons unsigned int
CONS of the command.
what string
What is wrong.

ArchMsg.Error.tlb_entries_overlap

A TLB entry was attempted to be inserted into the TLB and was determined that it overlaps an existing entry. This check is not perfect but will catch simple errors. Fields:

do_f_tlb_conflict bool
Chosen to perform an F_TLB_CONFLICT.
end_address_of_new_entry unsigned int
End address of new entry.
end_address_of_old_entry unsigned int
End address of old entry.
index_of_new_entry unsigned int
Index of new entry.
index_of_old_entry unsigned int
Index of old entry.
start_address_of_new_entry unsigned int
Start address of new entry.
start_address_of_old_entry unsigned int
Start address of old entry.
why enum
Why it is an error for these two entries to overlap.

ArchMsg.Error.tlb_entry_not_invalidated_due_to_ril

An entry in the cache was not invalidated even though in the right address range because of the RIL fields of the command the entry do not match. Fields:

cmd_num unsigned int
The NUM field of the RIL part of the command.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

cmd_ril_tg enum
: The RIL_TG field of the RIL part of the command.

cmd_ril_ttl unsigned int
: The TTL field of the RIL part of the command, zero means any level.

cmd_scale unsigned int
: The SCALE field of the RIL part of the command.

entry_id unsigned int
: The entry id that is being invalidated.

tlb_entry string
: The TLB entry.

ArchMsg.Error.vatos_sel_vmid_out_of_range

The SMMU_(S_)VATOS_SEL.VMID field was programmed with a VMID that was too wide for this implementation (SMMU_IDR0.VMID16 == 0). Fields:

ssd_ns bool
: The security state of the VATOS interface

vmid unsigned int
: The VMID programmed.

ArchMsg.Info.info

These are information messages about what is happening in the SMMU. Fields:

output string
: The stream output

ArchMsg.Warning.CMD_RESUME_no_transaction_resumed

A CMD_RESUME was issued that matched no transaction. Fields:

stag unsigned int
: STAG in the CMD_RESUME.

streamid unsigned int
: StreamID in the CMD_RESUME.

streamid_ns bool
: The StreamID was for the non-secure world.

ArchMsg.Warning.atc_inv_strange

Something was odd about the CMD_ATC_INV. Fields:

cmd_id unsigned int
: Command id

why string
: Why the CMD_ATC_INV was strange

ArchMsg.Warning.bad_conf_reset_of_SMMU_GBPA_ABORT

The signal was not driven at reset by the external system, but was driven after we first started using the default value we had assumed. The external system, if it is going to drive the signal, must do so at first reset so that both sides of the connection can agree on the value of this signal. Fields:

default_value bool
: The default value of the signal we have been assuming.

value bool
: The value of the signal.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.bad_conf_reset_of_SMMU_S_GBPA_ABORT

The signal was not driven at reset by the external system, but was driven after we first started using the default value we had assumed. The external system, if it is going to drive the signal, must do so at first reset so that both sides of the connection can agree on the value of this signal. Fields:

default_value bool
: The default value of the signal we have been assuming.

value bool
: The value of the signal.

ArchMsg.Warning.bad_conf_system_supports_httu

The signal was not driven at reset by the external system, but was driven after we first started using the default value we had assumed. The external system, if it is going to drive the signal, must do so at first reset so that both sides of the connection can agree on the value of this signal. Fields:

default_value bool
: The default value of the signal we have been assuming.

value bool
: The value of the signal.

ArchMsg.Warning.bad_sec_override

The signal was not driven at reset by the external system, but was driven after we first started using the default value we had assumed. The external system, if it is going to drive the signal, must do so at first reset so that both sides of the connection can agree on the value of this signal. Fields:

default_value bool
: The default value of the signal we have been assuming.

value bool
: The value of the signal.

ArchMsg.Warning.bad_sup_btm

The signal was not driven at reset by the external system, but was driven after we first started using the default value we had assumed. The external system, if it is going to drive the signal, must do so at first reset so that both sides of the connection can agree on the value of this signal. Fields:

default_value bool
: The default value of the signal we have been assuming.

value bool
: The value of the signal.

ArchMsg.Warning.bad_sup_cohacc

The signal was not driven at reset by the external system, but was driven after we first started using the default value we had assumed. The external system, if it is going to drive the signal, must do so at first reset so that both sides of the connection can agree on the value of this signal. Fields:

default_value bool
: The default value of the signal we have been assuming.

value bool
: The value of the signal.

ArchMsg.Warning.bad_sup_sev

The signal was not driven at reset by the external system, but was driven after we first started using the default value we had assumed. The external system, if it is going to drive the signal, must do so at first reset so that both sides of the connection can agree on the value of this signal. Fields:

default_value bool
: The default value of the signal we have been assuming.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

value bool
The value of the signal.

ArchMsg.Warning.bad_tbu0_reset_in

The signal was not driven at reset by the external system, but was driven after we first started using the default value we had assumed. The external system, if it is going to drive the signal, must do so at first reset so that both sides of the connection can agree on the value of this signal. Fields:

default_value bool
The default value of the signal we have been assuming.
value bool
The value of the signal.

ArchMsg.Warning.bad_tcu_reset_in

The signal was not driven at reset by the external system, but was driven after we first started using the default value we had assumed. The external system, if it is going to drive the signal, must do so at first reset so that both sides of the connection can agree on the value of this signal. Fields:

default_value bool
The default value of the signal we have been assuming.
value bool
The value of the signal.

ArchMsg.Warning.contig_bit_gives_too_large_region_for_TxSZ

If the contig bit was used then the size of the contig region would be larger than that indicated by TxSZ. Fields:

desc_kind enum
Descriptor kind
el enum
Exception level
input_address unsigned int
Input address
log2_contig_region_size unsigned int
Log2(contig region size in bytes)
output_address unsigned int
Output address
stage_and_level unsigned int
The stage in bits [7:4] and the level in [3:0].
vmid unsigned int
VMID if appropriate

ArchMsg.Warning.contig_bit_has_inconsistent_input_and_output_address

If the contig bit was used then the some bits of the output address held in the descriptor and the input address must match. Fields:

desc_kind enum
Descriptor kind
el enum
Exception level
input_address unsigned int
Input address
log2_contig_region_size unsigned int
Log2(contig region size in bytes)
match_mask unsigned int
Bits that must match.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

output_address unsigned int
: Output address

stage_and_level unsigned int
: The stage in bits [7:4] and the level in [3:0].

vmid unsigned int
: VMID if appropriate

ArchMsg.Warning.fetch_from_memory_type_not_supporting_httu

A descriptor fetch from an HTTU-enabled translation regime to an unsupported memory type was made. Whilst the fetch itself may succeed, if an update to the descriptor was attempted then it would fail. Fields:

address unsigned int
: The address of the descriptor fetch.

desc_inner enum
: Inner cacheability of descriptor

desc_outer enum
: Outer cacheability of descriptor

desc_prot enum
: The instruction/data, bus-ns/s and privileged/user marking

desc_sh enum
: Shareability of descriptor

ssd_ns bool
: Is the Security State Determination of the transaction non-secure.

stage unsigned int
: The stage at which we had a problem.

streamid unsigned int
: The StreamID of the transaction

substreamid unsigned int
: The SubstreamID of the transaction (or ~0u if no substreamid)

trans_id unsigned int
: Transaction id.

ArchMsg.Warning.msi_address_truncated

An MSI was generated, but the address was silently truncated due to the limited downstream address bus width. Fields:

pmcg_index_and_counter unsigned int
: If this is a PMCG interrupt, then the top 16 bits are the PMCG index, the lower 16 bits are counter index.

ras_error_group_id_node_id_and_record_index_in_node unsigned int
: If this is a RAS interrupt then this is the error group record id in bits[23:16], the node_id in [15:8] and the record index in the node in [7:0].

truncated_address unsigned int
: The truncated address.

untruncated_address unsigned int
: The untruncated address.

which enum
: Which MSI this is

ArchMsg.Warning.msi_lost

An MSI was attempted to be sent, but the implementation doesn't support it. Fields:

id unsigned int
: ID of this interrupt transaction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

kind enum
What kind of interrupt.

ArchMsg.Warning.pmcg_non_secure_world

A commentary on potential mis-programming of the PMCG when transitioning the PMCG from the secure world to the non-secure. Fields:

commentary string
The commentary.

ArchMsg.Warning.pmcg_programming_violates_security

A commentary on potential mis-programming of the PMCG when transitioning the PMCG between non-secure and secure worlds. This is under the assumption that a non-secure asynchronous agent could be writing to the PMCG. Fields:

commentary string
The commentary.

ArchMsg.Warning.priq_auto_response_failed_to_find_STE

The PRIQ was going to generate an auto-response, but failed to find an STE and so is returning a Failure message to the EndPoint which should disable the PRI interface of the EndPoint. Fields:

trans_id unsigned int
The ID of the PPR that is being lost.
why enum
Why the PPR is being lost.

ArchMsg.Warning.priq_overflow_bad_acking

Indicates that an overflow condition was acknowledged by writing to:- SMMU_PRIQ_CONS.OVACKFLG but an overflow condition did not exist. Fields:

new_ovackflg bool
The new value of the SMMU_PRIQ_CONS.OVACKFLG.

ArchMsg.Warning.priq_smmuen_forces_effective_priqen_low

If SMMUEN == 0, then the effective value of PRIQEN is 0. This warning is triggered when PRIQEN == 1 && SMMUEN == 0; which may not be what was intended. The PRIQ cannot be active if SMMUEN == 0.

ArchMsg.Warning.sev_lost

A SEV was lost because it isn't supported according to SMMU_IDR0.SEV. Fields:

why enum
Why the SEV was generated.

ArchMsg.Warning.smmu_pcie_rc_is_in_reset_ignoring_atc_invalidate

A CMD_PRI_RESP or CMD_ATC_INV was sent with a streamid that mapped to a PCIe Root Complex that is in reset. The message will be discarded and act as though completed successfully. Fields:

port_index unsigned int
The port index (node index) that is in reset but we would have sent it to.
streamid unsigned int
The StreamID that failed to map.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.smmu_pcie_rc_not_found_for_streamid

A CMD_PRI_RESP or CMD_ATC_INV was sent with a streamid that did not map to a PCIe Root Complex. This might be that the SW used an incorrect StreamID or it might be that the model has not been connected correctly. ATC Invalidate messages complete as though successful and PRI Requests are ignored. Fields:

streamid unsigned int
: The StreamID that failed to map.

ArchMsg.Warning.smmu_pmusnapshot

Something strange happened on the pmusnapshot_req/pmusnapshot_ack inteface. Fields:

pin_index unsigned int
: If the PMCG index corresponds to an array of signals, this is the index in the array, or 0 otherwise.

pmcg_index enum
: The PMCG index.

warning string
: The warning message.

ArchMsg.Warning.warning

These messages are about unusual (but not necessarily incorrect) activity occurring on the SMMU. Fields:

output string
: The stream output

EVENTQ_config

Configuration of interrupt updated. Fields:

address unsigned int
: Address of the MSI.

data unsigned int
: Data payload of the MSI.

irqen bool
: Appropriate bit of SMMU_s_IRQ_CTRL controlling this interrupt.

memattr enum
: Memory type

ns bool
: Non-secure bus attribute.

ra bool
: read-allocate attribute.

sh enum
: Shareability.

smmu_idr0_msi bool
: Are MSIs supported for this security world.

tr bool
: transient attribute

wa bool
: write-allocate attributes.

EVENTQ_irqen

The IRQEN has been changed. Fields:

new_value bool
: The new value of IRQEN.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

EVENTQ_irqen_ack

The IRQEN change has been acked. Fields:

new_value bool
: The new value of IRQEN.

GERROR_config

Configuration of interrupt updated. Fields:

address unsigned int
: Address of the MSI.

data unsigned int
: Data payload of the MSI.

irqen bool
: Appropriate bit of SMMU_s_IRQ_CTRL controlling this interrupt.

memattr enum
: Memory type

ns bool
: Non-secure bus attribute.

ra bool
: read-allocate attribute.

sh enum
: Shareability.

smmu_idr0_msi bool
: Are MSIs supported for this security world.

tr bool
: transient attribute

wa bool
: write-allocate attributes.

GERROR_irqen

The IRQEN has been changed. Fields:

new_value bool
: The new value of IRQEN.

GERROR_irqen_ack

The IRQEN change has been acked. Fields:

new_value bool
: The new value of IRQEN.

PRIQ_config

Configuration of interrupt updated. Fields:

address unsigned int
: Address of the MSI.

data unsigned int
: Data payload of the MSI.

irqen bool
: Appropriate bit of SMMU_s_IRQ_CTRL controlling this interrupt.

memattr enum
: Memory type

ns bool
: Non-secure bus attribute.

ra bool
: read-allocate attribute.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

sh enum
: Shareability.

smmu_idr0_msi bool
: Are MSIs supported for this security world.

tr bool
: transient attribute

wa bool
: write-allocate attributes.

PRIQ_irqen

The IRQEN has been changed. Fields:

new_value bool
: The new value of IRQEN.

PRIQ_irqen_ack

The IRQEN change has been acked. Fields:

new_value bool
: The new value of IRQEN.

SMMU_CR0ACK_SMMUEN_hazarded_by_priq

The SMMU_CR0ACK.SMMUEN cannot acknowledge the change to SMMUEN because there are outstanding PRIQ writes.

SMMU_CR0ACK_SMMUEN_update

The acknowledge to SMMU_CR0.SMMUEN was performed. Fields:

new_value bool
: The new value of the register.

SMMU_CR0_SMMUEN_old_set_complete

A set of transactions associated with the old value of SMMU_CR0.SMMUEN completed. Fields:

last bool
: This is the last set completing.

old_value bool
: The old value of the set completing.

SMMU_CR0_SMMUEN_write

A write to SMMU_CR0.SMMUEN occurred. Fields:

new_value bool
: The new value of the register.

old_value bool
: The old value of the register.

SMMU_GBPA_old_set_complete

A set of transactions associated with the old value of SMMU_GBPA completed. Fields:

last bool
: This is the last set completing.

old_value unsigned int
: The old value of the set completing.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SMMU_GBPA_update

The Update flag to SMMU_GBPA was lowered. Fields:

new_value unsigned int
: The new value of the register.

SMMU_GBPA_write

A write to SMMU_GBPA occurred. Fields:

new_value unsigned int
: The new value of the register.

old_value unsigned int
: The old value of the register.

SMMU_S_CR0ACK_SMMUEN_update

The acknowledge to SMMU_S_CR0.SMMUEN was performed. Fields:

new_value bool
: The new value of the register.

SMMU_S_CR0_SMMUEN_old_set_complete

A set of transactions associated with the old value of SMMU_S_CR0.SMMUEN completed. Fields:

last bool
: This is the last set completing.

old_value bool
: The old value of the set completing.

SMMU_S_CR0_SMMUEN_write

A write to SMMU_S_CR0.SMMUEN occurred. Fields:

new_value bool
: The new value of the register.

old_value bool
: The old value of the register.

SMMU_S_GBPA_old_set_complete

A set of transactions associated with the old value of SMMU_S_GBPA completed. Fields:

last bool
: This is the last set completing.

old_value unsigned int
: The old value of the set completing.

SMMU_S_GBPA_update

The Update flag to SMMU_S_GBPA was lowered. Fields:

new_value unsigned int
: The new value of the register.

SMMU_S_GBPA_write

A write to SMMU_S_GBPA occurred. Fields:

new_value unsigned int
: The new value of the register.

old_value unsigned int
: The old value of the register.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

S_EVENTQ_config

Configuration of interrupt updated. Fields:

address unsigned int
: Address of the MSI.

data unsigned int
: Data payload of the MSI.

irqen bool
: Appropriate bit of SMMU_s_IRQ_CTRL controlling this interrupt.

memattr enum
: Memory type

ns bool
: Non-secure bus attribute.

ra bool
: read-allocate attribute.

sh enum
: Shareability.

smmu_idr0_msi bool
: Are MSIs supported for this security world.

tr bool
: transient attribute

wa bool
: write-allocate attributes.

S_EVENTQ_irqen

The IRQEN has been changed. Fields:

new_value bool
: The new value of IRQEN.

S_EVENTQ_irqen_ack

The IRQEN change has been acked. Fields:

new_value bool
: The new value of IRQEN.

S_GERROR_config

Configuration of interrupt updated. Fields:

address unsigned int
: Address of the MSI.

data unsigned int
: Data payload of the MSI.

irqen bool
: Appropriate bit of SMMU_s_IRQ_CTRL controlling this interrupt.

memattr enum
: Memory type

ns bool
: Non-secure bus attribute.

ra bool
: read-allocate attribute.

sh enum
: Shareability.

smmu_idr0_msi bool
: Are MSIs supported for this security world.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

tr bool
: transient attribute

wa bool
: write-allocate attributes.

S_GERROR_irqen

The IRQEN has been changed. Fields:

new_value bool
: The new value of IRQEN.

S_GERROR_irqen_ack

The IRQEN change has been acked. Fields:

new_value bool
: The new value of IRQEN.

TLB.tlb_commentary

A commentary from the cache about what is going on. Fields:

output string
: The stream output.

TLB.tlb_entry_invalidated

An entry in the cache has been invalidated. Fields:

entry_id unsigned int
: The entry id that is being invalidated.

reason enum
: Why the entry is being invalidated.

atc_inv_nop

The CMD_ATC_INV command is ignored as a NOP. This may be emitted multiple times if the CMD_ATC_INV is being ignored for multiple reasons. Fields:

address unsigned int
: Untranslated Address to invalidate.

cmd_id unsigned int
: Command id, if top-bit is set then was issued from the Non-secure CMDQ.

cons unsigned int
: CONS of the command.

global bool
: Global flag.

log2_size_in_bytes unsigned int
: Log 2 of the size in bytes of the region to invalidate.

streamid unsigned int
: StreamID in the invalidate command.

substreamid unsigned int
: SubstreamID or ~0u if SSValid == false.

why enum
: Why was NOPped

atos_complete_fault

The ATOS operation completed with a fault. Fields:

effective_st1translate bool
: Because of the settings and/or the ATOS type then the effective st1translate can be different.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

fault_faddr unsigned int
: The fault FADDR.

fault_faultcode enum
: The fault code.

fault_reason enum
: The fault reason.

ssd_ns bool
: This is a non-secure ATOS operation.

st1translate bool
: The StreamID/SubstreamID combination should have been translated by stage 1.

st2translate bool
: The translation suffered a stage 2 translation.

state enum
: The final transaction state of the ATOS operation.

vatos_vmid unsigned int
: The VMID of the VATOS operation, or ~0u if not a VATOS operation.

atos_complete_fault_inv_req

The ATOS operation completed, faulted and generated an INV_REQ response. Fields:

ssd_ns bool
: This is a non-secure ATOS operation.

vatos_vmid unsigned int
: The VMID of the VATOS operation, or ~0u if not a VATOS operation.

why enum
: Why the ATOS operation generated an INV_REQ

atos_complete_success

The ATOS operation completed successfully. Fields:

base_addr unsigned int
: The actual base address of region.

effective_st1translate bool
: Because of the settings and/or the ATOS type then the effective st1translate can be different.

par_addr unsigned int
: The PAR.ADDR field.

par_mair unsigned int
: The memory attributes encoded as a MAIR

par_ns bool
: The PAR.NS field, for an SSD-ns request then this will always be 0.

par_sh enum
: Shareability

par_size bool
: The PAR.Size field.

size_in_bytes unsigned int
: The actual size in bytes of the region.

ssd_ns bool
: This is a non-secure ATOS operation.

st1translate bool
: The StreamID/SubstreamID combination should have been translated by stage 1.

st2translate bool
: The translation suffered a stage 2 translation.

state enum
: The final transaction state of the ATOS operation.

vatos_vmid unsigned int
: The VMID of the VATOS operation, or ~0u if not a VATOS operation.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

atos_run_set

The SMMU_s_GATOS_CTRL.RUN field was set to start the ATOS operation. Fields:

ssd_ns bool
: This is a non-secure ATOS operation.

vatos bool
: This is a VATOS operation

atos_starting

The ATOS operation is starting. Fields:

addr unsigned int
: The input address to the ATOS operation.

httui bool
: Inhibit HTTU update.

ind bool
: Instruction Data.

pnu bool
: Privileged not User.

rnw bool
: Read not Write.

ssd_ns bool
: This is a non-secure ATOS operation.

ssec bool
: If this is a secure ATOS operation then this is if it is secure or not.

streamid unsigned int
: The StreamID requested.

substreamid unsigned int
: The SubstreamID requested, or ~0u if no SubstreamID.

type enum
: The requested ATOS type.

vatos_vmid unsigned int
: The VMID of the VATOS operation, or ~0u if not a VATOS operation.

cd_cc.CD_commentary

A commentary from the cache about what is going on. Fields:

output string
: The stream output.

cd_cc.CD_entry_invalidated

An entry in the cache has been invalidated. Fields:

entry_id unsigned int
: The entry id that is being invalidated.

reason enum
: Why the entry is being invalidated.

cd_entry_allocated

An CD entry has been allocated. Fields:

cd string
: A textual description of the CD.

entry_id unsigned int
: The id of the cache entry. Top bit is set if is a debug entry.

ns enum
: For the non-secure world

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

streamid unsigned int
: The StreamID of the entry this will match.

substreamid unsigned int
: The SubstreamID of the entry this will match. This may be zero for transactions without a SubstreamID.

conf_reset_of_SMMU_GBPA_ABORT

The reset value of SMMU_GBPA.ABORT. Fields:

value bool
: The value of the signal.

conf_reset_of_SMMU_S_GBPA_ABORT

The reset value of SMMU_S_GBPA.ABORT. Fields:

value bool
: The value of the signal.

conf_system_supports_httu

The system supports HTTU. Fields:

value bool
: The value of the signal.

dvm_tlbinvalidate_complete

The DVM TLB Invalidate message completed. Fields:

id unsigned int
: The unique id of this DVM message.

ok bool
: The DVM message was OK.

dvm_tlbinvalidate_received

A DVM message for a TLB Invalidate has been received. Fields:

address unsigned int
: The VA or IPA to use if match_address.

asid unsigned int
: The ASID to match if match_asid.

by_ipa bool
: The operation is for an IPA operation if match_address.

id unsigned int
: The unique id of this DVM message.

ignored enum
: The DVM message was ignored

last_level bool
: The operation is for last level if supported.

match_address bool
: Match the address field.

match_asid bool
: Match the asid field.

match_vmid bool
: Match the vmid field.

num unsigned int
: If a range operation, the NUM field. If a single-address operation this is 0.

prot enum
: The protection level for which this TLB Invalidate will operate on.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

security_world enum
: The security world that this will apply to.

smmu_scale unsigned int
: If a range operation, then the SCALE field with the meaning in the SMMU architecture which is different to the PE architecture. If a single-address operation this is 0.

stage1_only bool
: The operation is for stage 1 only if supported.

tg enum
: If a single-address or address-range operation, then the Translation Granule hint. Address-range operations always supply a Translation Granule.

translation_table_level enum
: The leaf level of the translation table.

vmid unsigned int
: The VMID to match if match_vmid.

found_tlb_entry_has_different_aset

Architecturally, a particular ASID either should be ASET0 or ASET1. However, we have managed to find a TLB entry that has a different ASET than that which we were searching for. This indicates a programming error. You should examine all contexts with this particular ASID/VMID and ensure they are consistent. Fields:

aset unsigned int
: The ASET of this entry.

asid unsigned int
: ASID if appropriate.

index unsigned int
: Index of TLB entry.

input_start_address unsigned int
: The start address of the input range that this matches.

scheme enum
: The tagging scheme used

ssd_ns enum
: Security State

vmid unsigned int
: VMID if appropriate.

gerror_inverted

A GERROR bit was attempted to be inverted to record a fault. Fields:

already_different_to_gerrorn bool
: True if the bit was already different to the corresponding bit in GERRORN and so it was not inverted.

gerrorn unsigned int
: GERRORN register value.

interrupt_potentially_sent bool
: An interrupt is potentially sent, depending on IRQEN.
GERROR(N).MSI_GERROR_ABT_ERR does not send an interupt.

new_gerror unsigned int
: GERROR register value after any invertion.

which enum
: Which bit was attempted to be inverted.

gerrorn_acknowledge

A GERROR was acknowledged by SW. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

which enum
: Which GERROR was acknowledged.

httu_update_abandoned_update

The HTTU update of a descriptor in memory was potentially possible, but it was behind an update that failed to apply cleanly. Fields:

AF enum
: Whether an AF update should/could occur.

DBM enum
: Whether a DBM update should/could occur.

address unsigned int
: Address of HTTU update.

httu_inner enum
: Inner cacheability of descriptor to update

httu_outer enum
: Outer cacheability of descriptor to update

httu_prot enum
: The instruction/data, bus-ns/s and privileged/user marking

httu_sh enum
: Shareability of descriptor to update

mpam_pmg_and_partid unsigned int
: Bits[23:16] are the MPAM PMG. Bits[15:0] are the physical PARTID.

original_descriptor unsigned int
: The original descriptor that the model observed.

stage_and_level unsigned int
: The stage (bits[7:4]) and level (bits[3:0]).

trans_id unsigned int
: The transaction id that this HTTU update is associated with.

what enum
: What this descriptor represents

httu_update_end_update

The attempted update of a descriptor in memory has occurred. Fields:

is_big_endian bool
: The descriptor is big-endian in memory.

original_descriptor unsigned int
: The original descriptor value.

result enum
: The result of the attempt to update.

trans_id unsigned int
: The transaction id that this HTTU update is associated with.

try_to_change_to_descriptor unsigned int
: The new descriptor value that tried to replace the original.

value_that_was_in_memory unsigned int
: The value that the compare-and-swap operation returned as the value that was in memory.

httu_update_not_done

A discretionary HTTU update could occur and the implementation choose not to do it. A mandatory_do_if_linked_performed means that this is a stage 2 leaf descriptor that maps a subsequent stage 1 leaf descriptor whose update is discretionary and if that subsequent discretionary update is going to occur then this update becomes mandatory. Fields:

AF enum
: Whether an AF update should/could occur.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

DBM enum
: Whether a DBM update should/could occur.

address unsigned int
: Address of HTTU update.

httu_inner enum
: Inner cacheability of descriptor to update

httu_outer enum
: Outer cacheability of descriptor to update

httu_prot enum
: The instruction/data, bus-ns/s and privileged/user marking

httu_sh enum
: Shareability of descriptor to update

mpam_pmg_and_partid unsigned int
: Bits[23:16] are the MPAM PMG. Bits[15:0] are the physical PARTID.

original_descriptor unsigned int
: The original descriptor that the model observed.

stage_and_level unsigned int
: The stage (bits[7:4]) and level (bits[3:0]).

trans_id unsigned int
: The transaction id that this HTTU update is associated with.

what enum
: What this descriptor represents

httu_update_start_update

An HTTU update could occur and the implementation chose to try it. A mandatory_do_if_linked_performed means that this is a stage 2 leaf descriptor that maps a subsequent stage 1 leaf descriptor whose update is discretionary and if that subsequent discretionary update is going to occur then this update becomes mandatory. Fields:

AF enum
: Whether an AF update should/could occur.

DBM enum
: Whether a DBM update should/could occur.

address unsigned int
: Address of HTTU update.

httu_inner enum
: Inner cacheability of descriptor to update

httu_outer enum
: Outer cacheability of descriptor to update

httu_prot enum
: The instruction/data, bus-ns/s and privileged/user marking

httu_sh enum
: Shareability of descriptor to update

is_big_endian bool
: The descriptor will be written to memory as big-endian.

mpam_pmg_and_partid unsigned int
: Bits[23:16] are the MPAM PMG. Bits[15:0] are the physical PARTID.

original_descriptor unsigned int
: The original descriptor that the model observed.

stage_and_level unsigned int
: The stage (bits[7:4]) and level (bits[3:0]).

trans_id unsigned int
: The transaction id that this HTTU update is associated with.

try_to_change_to_descriptor unsigned int
: The new descriptor value that is going to try replace the original.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

what enum
: What this descriptor represents

will_do_AF bool
: What the implementation chose to do for the AF update.

will_do_DBM bool
: What the implementation chose to do for the DBM update.

interrupt_returned

An interrupt/MSI returned from downstream. Fields:

InD_NS_PnU enum
: Instruction/Data, Non-secure/Secure, Privileged/User.

address unsigned int
: Address of the MSI.

data unsigned int
: Data payload of the MSI.

id unsigned int
: ID of this interrupt transaction.

inner_cache enum
: The actual attributes of the access.

kind enum
: What kind of interrupt.

mpam_ns bool
: The security state associated with the MPAM PARTID/PMG.

mpam_pmg_and_partid unsigned int
: Bits[23:16] are the MPAM PMG. Bits[15:0] are the physical PARTID.

msi_valid bool
: MSI will attempt to be sent.

ok bool
: Did the access return OK or an abort?

outer_cache enum
: The actual attributes of the access.

pmcg_index_and_counter unsigned int
: If this is a PMCG interrupt, then the top 16 bits are the PMCG index, the lower 16 bits are counter index.

ras_error_group_id_node_id_and_record_index_in_node unsigned int
: If this is a RAS interrupt then this is the error group record id in bits[23:16], the node_id in [15:8] and the record index in the node in [7:0].

sh enum
: Shareability

interrupt_sent

An interrupt is raised. If it sends an MSI then this is _after_ any device-dependent transform on the architectural attributes and so may differ from what is programmed. Fields:

InD_NS_PnU enum
: Instruction/Data, Non-secure/Secure, Privileged/User.

address unsigned int
: Address of the MSI.

data unsigned int
: Data payload of the MSI.

id unsigned int
: ID of this interrupt transaction.

inner_cache enum
: The actual attributes of the access.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

kind enum
: What kind of interrupt.

mpam_ns bool
: The NS state of the MPAM PARTID and MPAM PMG.

mpam_pmg_and_partid unsigned int
: Bits[23:16] are the MPAM PMG. Bits[15:0] are the physical PARTID.

msi_valid bool
: MSI will attempt to be sent.

outer_cache enum
: The actual attributes of the access.

pmcg_index_and_counter unsigned int
: If this is a PMCG interrupt, then the top 16 bits are the PMCG index, the lower 16 bits are counter index.

ras_error_group_id_node_id_and_record_index_in_node unsigned int
: If this is a RAS interrupt then this is the error group record id in bits[23:16], the node_id in [15:8] and the record index in the node in [7:0].

sh enum
: Shareability

l1cd_cc.L1CD_commentary

A commentary from the cache about what is going on. Fields:

output string
: The stream output.

l1cd_cc.L1CD_entry_invalidated

An entry in the cache has been invalidated. Fields:

entry_id unsigned int
: The entry id that is being invalidated.

reason enum
: Why the entry is being invalidated.

l1cd_entry_allocated

An L1 CD entry has been allocated. Fields:

entry_id unsigned int
: The id of the cache entry. Top bit is set if is a debug entry.

normalised_substreamid unsigned int
: The first SubstreamID of the range of SubstreamIDs that this L1CD entry will match.

ns enum
: For the non-secure world

pa_l2 unsigned int
: The PA of the L2 CD table. This is L2Ptr << 12.

streamid unsigned int
: The StreamID this CD is for.

valid bool
: Is the entry valid.

l1ste_cc.L1STE_commentary

A commentary from the cache about what is going on. Fields:

output string
: The stream output.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

l1ste_cc.L1STE_entry_invalidated

An entry in the cache has been invalidated. Fields:

entry_id unsigned int
: The entry id that is being invalidated.

reason enum
: Why the entry is being invalidated.

l1ste_entry_allocated

An L1 STE entry has been allocated. Fields:

entry_id unsigned int
: The id of the cache entry. Top bit is set if is a debug entry.

first_streamid_of_range unsigned int
: The first StreamID of the range of StreamIDs that this L1STE entry will match.

ns enum
: For the non-secure world

num_entries_in_l2 unsigned int
: The number of entries in the L2 table. This is 2**(Span-1) or 0 if invalid.

pa_l2 unsigned int
: The PA of the L2 ST table. This is L2Ptr << 6 and aligned to the size of the table.

ns_cmd_sync_completed_irq

Non-secure CMD_SYNC completed its IRQ action. Fields:

cmd_id unsigned int
: The ID of the command being issued.

cons_incl_wrap unsigned int
: CONS pointer of CMD_SYNC.

ns_cmd_sync_completed_no_action

Non-secure CMD_SYNC completed, there was no CMD_SYNC action (SIG_NONE). Fields:

cmd_id unsigned int
: The ID of the command being issued.

cons_incl_wrap unsigned int
: CONS pointer of CMD_SYNC.

ns_cmd_sync_completed_sev

Non-secure CMD_SYNC completed its action of SIG_SEV. Fields:

cmd_id unsigned int
: The ID of the command being issued.

cons_incl_wrap unsigned int
: CONS pointer of CMD_SYNC.

ns_cmd_sync_completed_with_error

Non-secure CMD_SYNC completed, however, there was an error associated with the completion of the CMD_SYNC. Fields:

cerror enum
: The error associated with the CMD_SYNC

cmd_id unsigned int
: The ID of the command being issued.

cons_incl_wrap unsigned int
: CONS pointer of CMD_SYNC.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ns_cmd_sync_issued

Non-secure CMD_SYNC has now completed issuing and we now know if something is hazarding the CMD_SYNC from completing. Fields:

cmd_id unsigned int
: The ID of the command being issued.

cons_incl_wrap unsigned int
: CONS pointer of CMD_SYNC.

hazarded bool
: The CMD_SYNC is currently hazarded from completing.

ns_cmd_sync_issuing

Non-secure CMD_SYNC is starting to issue, we are going to start accounting for which things have to be part of the prior set of things that have to complete before the CMD_SYNC can complete. Fields:

cmd_id unsigned int
: The ID of the command being issued.

cons_incl_wrap unsigned int
: CONS pointer of CMD_SYNC.

ns_cmd_sync_starting_completion_action_irq

Non-secure CMD_SYNC starting IRQ completion action. Fields:

cmd_id unsigned int
: The ID of the command being issued.

cons_incl_wrap unsigned int
: CONS pointer of CMD_SYNC.

msiaddress unsigned int
: The MSI address to use (or 0 if not sending an MSI).

msidata unsigned int
: The MSI data to send.

ns_cmd_sync_starting_completion_action_sev

Non-secure CMD_SYNC starting its completion action of SIG_SEV. Fields:

cmd_id unsigned int
: The ID of the command being issued.

cons_incl_wrap unsigned int
: CONS pointer of CMD_SYNC.

ns_cmdq_cmd_sync_error

The CMD_SYNC can fault if an ATC Invalidate timeout occurs. The timeout will be attached to the next CMD_SYNC on the queue on which the invalidate was launched. Fields:

cerror enum
: The CERROR to attach.

cmd_queue unsigned int
: The top bits indicate the CMDQ id. These bits are attached to all cmd_ids generated for this queue.

ns_cmdq_cmd_sync_error_delivered

The CMD_SYNC can fault if an ATC Invalidate timeout occurs. This is the CMD_SYNC delivering that fault. Fields:

cerror enum
: The CERROR to attach.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

cmd_id unsigned int
: Command ID of the CMD_SYNC

ns_cmdq_ecmdq_enack

Trace the changes to ENACK. Fields:

cerror enum
: The error being exposed, if any.

cmd_queue unsigned int
: The top bits indicate the CMDQ id. These bits are attached to all cmd_ids generated for this queue.

enack bool
: The value of ENACK.

forced_error bool
: The queue was enabled with ERR != ERRACK and so the implementation chooses to trigger the error reporting mechanism.

ns_cmdq_issue

A command is actually being executed. Fields:

cmd_id unsigned int
: Command ID of the command being executed.

cons unsigned int
: CONS of the command.

what string
: What is being executed.

ns_cmdq_pointers_state

The command queue pointers. Fields:

cmd_queue unsigned int
: The top bits indicate the CMDQ id. These bits are attached to all cmd_ids generated for this queue.

disable_fetch bool
: The model has decided that as you programmed PROD/CONS inconsistently that it would not fetch as it has no confidence as to which are commands and which are not.

q_issuable_and_issued_cons unsigned int
: The CONS pointer of the region that is issuable and/or issued (including wrap).

q_issuable_and_issued_prod unsigned int
: The PROD pointer of the region that is issuable and/or issued (including wrap).

q_programmer_cons unsigned int
: The CONS pointer as seen by the programmer (including wrap).

q_programmer_prod unsigned int
: The PROD pointer as seen by the programmer (including wrap).

q_unfetched_cons unsigned int
: The CONS pointer of the region unfetched (including wrap).

q_unfetched_prod unsigned int
: The PROD pointer of the region unfetched (including wrap).

ns_cmdq_state

A description of what the CMDQ can do now. Fields:

can_cerror bool
: There is a CERROR that can be recognised.

can_cmd_sync bool
: There is a CMD_SYNC that can be recognised.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

can_fetch bool
: The SMMU can fetch

can_issue bool
: There are fetched commands that are eligible to be issued.

can_update_cons bool
: The programmer view of CMDQ_CONS can be moved to say some have been consumed.

cmd_queue unsigned int
: The top bits indicate the CMDQ id. These bits are attached to all cmd_ids generated for this queue.

cmd_sync enum
: Why a CMD_SYNC cannot be issued

cmdqen bool
: CMDQEN, when enabled then the CMDQ can fetch commands

cmdqen_1_to_0_unacknowledged bool
: CMDQEN has been set to 0, but we have yet to acknowledge it.

current-real-cerror enum
: The current CERROR that would be visible to the programmer if there was an error.

fetch-disabled_due_to_programmer_error bool
: The PROD/CONS pointer was inconsistent and the model decided to stop processing commands.

fetch-number_of_unfetched_commands unsigned int
: The number of commands that have yet to be fetched from memory, or ~0u if CMDQEN == 0.

fetch-state enum
: The state of the fetch state machine

fetch-too_many_commands_outstanding_so_not_fetching_any_more bool
: If there are too may issued commands, or pending to be issued then we won't fetch any more.

issue-number_of_issuable_commands unsigned int
: The number of commands that have been fetched but not yet issued.

issue-number_of_unissuable_commands unsigned int
: This might include commands behind an illegal command, or behind a CMD_SYNC. Under some circumstances this can include a command representing an external abort.

issue-state enum
: The internal issued state machine state

outstanding_fetches_since_reset unsigned int
: The number of outstanding fetches.

ns_eventq_adding_to_cmd_sync_prior_set

The event created by the specified transaction id must become visible to the programmer before the CMD_SYNC can complete. Fields:

trans_id unsigned int
: The transaction id that caused the event.

ns_eventq_auto_retry_stalled_transaction

A transaction that was stalled but not yet reported on the Event Queue was retried. The retry can happen if the transaction didn't write because it was de-duplicated behind a reported stalling transaction and that stalling transaction was CMD_RESUMEd, or if the queue was unwriteable and the queue is now writeable. Fields:

trans_id unsigned int
: Transaction id of the auto-retried transaction.

why enum
: Why the transaction retried.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ns_eventq_cmd_sync_unhazarded

The CMD_SYNC has been unhazarded as the prior set is empty and as much of it as required is visible in the programmers' view. Fields:

prod_incl_wrap unsigned int
: The prod index including wrap bit that was required to become visible.

prog_prod_incl_wrap unsigned int
: The current programmer-visible prod including wrap bit.

ns_eventq_eventqen_and_eventqenack

Trace the values of EVENTQEN and EVENTQENACK as they change. Fields:

about enum
: Some more information about the queue state.

cons_incl_wrap unsigned int
: The CONS pointer (including wrap bits).

eventqen bool
: EVENTQEN.

eventqenack bool
: EVENTQENACK.

prod_incl_wrap unsigned int
: The PROD pointer (including wrap bits).

ns_eventq_external_abort

A particular eventq record got an external abort. The event will subsequently appear in the ns/s_eventq_lost_event_records trace. Fields:

prod unsigned int
: The prod pointer (excluding wrap bit) of the event record that aborted.

trans_id unsigned int
: The transaction id that produced the event record that aborted.

ns_eventq_lost_event_records

Event records that are lost and never record appear on this trace. Fields:

dword0 unsigned int
: Double word 0 of event.

dword1 unsigned int
: Double word 1 of event.

dword2 unsigned int
: Double word 2 of event.

dword3 unsigned int
: Double word 3 of event.

event enum
: Event code that was thrown away.

prod unsigned int
: The PROD index of the event if appropriate or ~0u otherwise.

reason enum
: The reason the record was lost.

trans_id unsigned int
: The transaction id that this corresponds to.

ns_eventq_overflow_acknowledged

When events are lost then they appear on the eventq_lost_event_records_* trace. This will activate the overflow flag in the corresponding SMMU_s_EVENTQ_PROD register. Software can then acknowledge that flag using the overflow acknowledge flag in SMMU_s_EVENTQ_CONS. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

bad bool
: If true then the programmer has attempted to acknowledge an overflow condition that didn't exist.

count_of_events_lost_due_to_overflow unsigned int
: This is the number of events that were lost because the event queue was full. This does not include any lost because of an EVENTQ_ABT_ERR.

ns_eventq_pending_event_records

Event records that are pending to be dealt with by the eventq state machine. A specific event can be traced multiple times if it becomes part of the prior set of transactions of a CMD_SYNC that has to complete before the CMD_SYNC is allowed to be completed. Fields:

dword0 unsigned int
: Double word 0 of event.

dword1 unsigned int
: Double word 1 of event.

dword2 unsigned int
: Double word 2 of event.

dword3 unsigned int
: Double word 3 of event.

event enum
: Event code.

is_prior_set bool
: The transaction forms part of the prior set of a CMD_SYNC that have to become visible (or thrown away if necessary) before the CMD_SYNC can complete

trans_id unsigned int
: The transaction id that this corresponds to.

ns_eventq_prod_cons_updated

This records when the programmer visible PROD/CONS pointer is updated. More can be visible in memory than are advertised in the programmer-visible PROD pointer. This also traces the latest PROD that could have been advertised. Fields:

mem_prod unsigned int
: The PROD pointer value for all records that are visible in the memory.

num_events_available unsigned int
: The number of events available to the programmer.

prog_cons unsigned int
: The programmer-visible CONS pointer.

prog_prod unsigned int
: The programmer-visible PROD pointer.

what enum
: What changed?

ns_eventq_raising_eventq_abt_err

EVENTQ_ABT_ERR can now be raised.

ns_eventq_removing_from_cmd_sync_prior_set

The event captured by a CMD_SYNC and created by the specified transaction id is now in memory. Fields:

prod unsigned int
: The prod index (no wrap bit) that must become visible to the programmer to see this record. If not head_of_line then the record will not become visible and this field is meaningless.

reason enum
: The reason the event was removed.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

trans_id unsigned int
The transaction id that caused the event.

ns_eventq_stashing_unreported_stalled_transaction_for_auto_retry

A stall event record would have been lost, instead it is stashed away to auto-retry when it can. Fields:

dword0 unsigned int
Double word 0 of event.
dword1 unsigned int
Double word 1 of event.
dword2 unsigned int
Double word 2 of event.
dword3 unsigned int
Double word 3 of event.
event enum
Event code that was thrown away.
reason enum
The reason the record was lost.
trans_id unsigned int
The transaction id that this corresponds to.

ns_eventq_write_event

An event is committed to being written. Fields:

dword0 unsigned int
Double word 0 of event.
dword1 unsigned int
Double word 1 of event.
dword2 unsigned int
Double word 2 of event.
dword3 unsigned int
Double word 3 of event.
event enum
Event code that was thrown away.
prod unsigned int
The PROD pointer (without wrap) that this event will be written to.
stall_event bool
Describes an event for a stalled transaction.
trans_id unsigned int
Transaction ID that generated this event.

ns_eventq_write_event_visible_in_memory

An event is now visible in memory. Fields:

abort bool
The write aborted.
dword0 unsigned int
Double word 0 of event.
dword1 unsigned int
Double word 1 of event.
dword2 unsigned int
Double word 2 of event.
dword3 unsigned int
Double word 3 of event.
event enum
Event code that was thrown away.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

prod unsigned int
: The PROD pointer (without wrap) that this event will be written to.

stall_event bool
: Describes an event for a stalled transaction.

trans_id unsigned int
: Transaction ID that generated this event.

pmcg_irq_config

The interrupt configuration of the Performance Monitor Counter Group (PMCG) changed. Fields:

address unsigned int
: Address of the MSI.

data unsigned int
: Data payload of the MSI.

irqen bool
: SMMU_PMCG_CTRL.IRQEN.

memattr enum
: Memory type

mpam_ns bool
: The NS state of the MPAM PARTID and MPAM PMG

mpam_pmg_and_partid unsigned int
: Bits[23:16] are the MPAM PMG. Bits[15:0] are the physical PARTID.

msi_supported bool
: Are MSIs supported for this security world.

ns bool
: Non-secure bus attribute.

number_of_interrupts_in_flight unsigned int
: The number of interrupts that have been committed to be produced or in flight.

pmcg_index unsigned int
: Index of the PMCG

sh enum
: Shareability.

smmu_pmcg_gmpam_Update bool
: The SMMU_PMCG_GMPAM.Update flag. Only when this is zero are writes predictable.

ssd_ns bool
: True if this PMCG controlled by the non-secure world.

pmcg_irqen

A trace of SMMU_PMCG_IRQ_CTRL.IRQEN. Fields:

new_value bool
: The new value of IRQEN.

pmcg_index unsigned int
: Index of the PMCG.

ssd_ns bool
: True if this PMCG controlled by the non-secure world.

pmcg_irqen_ack

A trace of SMMU_PMCG_IRQ_CTRLACK.IRQEN. Fields:

new_value bool
: The new value of IRQEN.

pmcg_index unsigned int
: Index of the PMCG.

ssd_ns bool
: True if this PMCG controlled by the non-secure world.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

pmcg_merging_interrupts

An interrupt was wanted to be generated, but one was already pending so the two were merged together. Fields:

pmcg_index unsigned int
: Index of the PMCG.

ssd_ns bool
: True if this PMCG controlled by the non-secure world.

pmcg_trigger

A PMCG counter has been triggered. Fields:

counter_index unsigned int
: The index of the counter within the PMCG.

event_id unsigned int
: The event id that has been triggered.

ns_event bool
: The security state associated with the event.

pmcg_index unsigned int
: Index of the PMCG.

streamid unsigned int
: The StreamID associated with the event, if there is one.

pmu_active_counter

Traces what active counters are in a PMCG and what streamids it might filter on. Those counters that trace both secure and non-secure StreamIDs or those that are not filtered by StreamID will appear twice, once for each security world. All active counters for a PMCG are traced one after another. Fields:

begin_streamid unsigned int
: The start StreamID to filter on.

counter_index unsigned int
: The counter index within the PMCG.

end_incl_streamid unsigned int
: The end inclusive StreamID to filter on.

evcnt unsigned int
: The current count.

event_id unsigned int
: The event ID to filter.

ns bool
: Are the StreamIDs non-secure?

pmcg_index unsigned int
: The index of the PMCG that this counter belongs to.

pmu_all_counters_in_pmcg_became_inactive

The PMCG was tracing some events and now is not tracing any. Fields:

pmcg_index unsigned int
: The index of the PMCG.

pmu_capture

For some reason, a capture event occurred. Fields:

pmcg_index unsigned int
: The index of the PMCG that the capture occurred on.

why enum
: Why did the capture occur?

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

pmu_counter_configured_to_use_unsupported_event

An enabled counter was configured to use a unsupported event. Fields:

counter_index unsigned int
: The counter index within the PMCG.

event_id unsigned int
: The unsupported event id.

pmcg_index unsigned int
: The index of the PMCG that this counter belongs to.

pmu_counter_overflowed

A counter in a particular PMCG overflowed. Fields:

already_overflowed bool
: True if the overflow flag was already set.

capture bool
: True if it captured the other counter values.

counter_index unsigned int
: The counter index within the PMCG.

interrupt bool
: True if going to attempt to generate an interrupt.

interrupt_action enum
: The interrupt action that is going to occur

pmcg_index unsigned int
: The index of the PMCG that this counter belongs to.

ssd_ns bool
: The PMCG is controlled by the Non-secure world.

priq_auto_response

A PRIQ auto response is generated. Fields:

prgindex unsigned int
: The PRG Index of the response.

resp enum
: The response to send.

streamid unsigned int
: The StreamID of the response.

substreamid unsigned int
: The PASID/SubstreamID attached to the response, or ~0u if there is none.

trans_id unsigned int
: The ID of the PPR that is being lost.

why enum
: Why the PPR is being lost.

priq_lost_ppr

The PRIQ is not reporting a PPR into the queue. It may or may not generate an auto-response. Fields:

is_stop_marker bool
: The PPR being lost is a Stop Marker.

resp enum
: What, if any, auto-response will be.

trans_id unsigned int
: The ID of the PPR that is being lost.

why enum
: Why the PPR is being lost

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

priq_overflow_acking

Indicates that an overflow condition was acknowledged by writing to:- SMMU_PRIQ_CONS.OVACKFLG. Fields:

new_ovackflg bool
: The new value of the SMMU_PRIQ_CONS.OVACKFLG.

priq_overflow_asserting

Indicates that we are toggling the SMMU_PRIQ_PROD.OVFLG because we lost a PRI request due to the PRIQ being full and an existing overflow condition does not already exist. Fields:

new_ovflg bool
: The new value of the SMMU_PRIQ_PROD.OVFLG.

trans_id unsigned int
: The transaction ID of the PPR that caused the overflow.

priq_received

A PRIQ Request has been received and is queued waiting processing. This does not mean that it has been written to the PRIQ but has been placed in a pending queue awaiting a decision about what to do about it. Fields:

L bool
: The Last in the Page Request Group (PRG).

R bool
: Read permission is requested.

W bool
: Read permission is requested.

prgindex unsigned int
: The Page Request Group (PRG) index that is being used to identify this request to the EndPoint.

ssd_ns bool
: The SSD of the StreamID. PRIQ should only receive _non-secure_ PRI Requests.

stop_marker bool
: If LRW == 0b100 then this indicates this is a stop marker.

streamid unsigned int
: The StreamID this request is associated with.

substreamid unsigned int
: The SubstreamID (PASID) this request is associated with, of ~0u if no SubstreamID.

substreamid_P bool
: If has a SubstreamID (!= ~0u) then Privileged Mode Requested.

substreamid_X bool
: If has a SubstreamID (!= ~0u) then Execution Requested (R must also be 1).

trans_id unsigned int
: The transaction ID that identifies this request.

untranslated_address unsigned int
: The Untranslated Address that this is requesting.

priq_write_aborted

A PRIQ write aborted. The PRIQ now goes into an error state and will start auto-responding to PRI requests. Fields:

prod_incl_wrap unsigned int
: The PROD including the Wrap bit where we were trying to write.

trans_id unsigned int
: The transaction ID of the PPR that aborted.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

priq_write_ok

A PRIQ write completed OK. Fields:

prod_incl_wrap unsigned int
: The PROD including the Wrap bit where we wrote..

trans_id unsigned int
: The transaction ID of the PPR.

priq_write_start

A PRIQ request has been received and is going to be attempt to be written to the queue. Fields:

L bool
: The Last in the Page Request Group (PRG).

R bool
: Read permission is requested.

W bool
: Read permission is requested.

prgindex unsigned int
: The Page Request Group (PRG) index that is being used to identify this request to the EndPoint.

prod_incl_wrap unsigned int
: PROD position including the wrap bit.

stop_marker bool
: If LRW == 0b100 then this indicates this is a stop marker.

streamid unsigned int
: The StreamID this request is associated with.

substreamid unsigned int
: The SubstreamID (PASID) this request is associated with, of ~0u if no SubstreamID.

substreamid_P bool
: If has a SubstreamID (!= ~0u) then Privileged Mode Requested.

substreamid_X bool
: If has a SubstreamID (!= ~0u) then Execution Requested (R must also be 1).

trans_id unsigned int
: The transaction ID that identifies this request.

untranslated_address unsigned int
: The Untranslated Address that this is requesting.

priqen_state

The trace of various fields that indicate the state of the PRIQ. Fields:

cons_incl_wrap unsigned int
: The value of SMMU_PRIQ.CONS.RD_and_RD_wrap.

number_of_pprs unsigned int
: The number of PPRs as indicated by the CONS/PROD.

number_of_pprs_still_to_deal_with unsigned int
: This is the number of PPRs that are currently waiting to either be written to the PRIQ, or auto-responded to.

number_of_priq_writes_in_flight unsigned int
: The number of writes to the PRIQ that are currently in flight.

ovackflg bool
: The OVACKFLG which if different to OVFLG is used to indicate that the PRIQ overflowed.

ovflg bool
: The OVFLG which if different to OVACKFLG is used to indicate that the PRIQ overflowed.

priq_abt_err bool
: There is an active SMMU_GERROR{N}.PRIQ_ABT_ERR.

priqen bool
: The value of SMMU_CR0.PRIQEN. The _effective_ value is 0 if SMMUEN == 0.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

priqenack bool
: The value of SMMU_CR0ACK.PRIQEN.

prod_incl_wrap unsigned int
: The value of SMMU_PRIQ.CONS.WR_and_WR_wrap.

queue_disabled_due_to_prior_programming_error bool
: The queue was disabled as the programmer got CONS/PROD into an inconsistent state. The model will disable the PRIQ until SW disables and re-enables the queue via SMMU_CR0.PRIQEN.

smmuen bool
: The value of SMMU_CR0.SMMUEN. If this is 0 then the effective PRIQEN is 0.

smmuenack bool
: The value of SMMU_CR0.SMMUEN. If this is 0 then the effective PRIQEN is 0.

table_size_in_elements unsigned int
: The size of the table in the number of items it can hold.

ptw_read

Page Table Walk (read). This is the result of the physical access that the SMMU is making. Fields:

abort enum
: Non-zero if the access aborted/failed.

adomain enum
: The actual attributes of the access that was used, after IMP DEF mangling.

aprot enum
: The actual attributes of the access that was used, after IMP DEF mangling.

data unsigned int
: The data fetch if it didn't abort.

desckind enum
: The kind of descriptor

inner_cache enum
: The actual attributes of the access that was used, after IMP DEF mangling.

input_address unsigned int
: Input address this lookup is for (after TBI normalisation).

ns bool
: Non-Secure

outer_cache enum
: The actual attributes of the access that was used, after IMP DEF mangling.

pa_address unsigned int
: Physical address.

ssd_ns enum
: Security state of stream

streamid unsigned int
: The StreamID this is for.

substreamid unsigned int
: The SubstreamID this is for.

trans_id unsigned int
: Transaction id

ttb_grain_stage_and_level unsigned int
: Which TTB used in bits[31:24]. Which Grain size used in bit[23:16]:- 0 -- 4 KiB 1 -- 16 KiB 2 -- 64 KiB 3 -- 64 KiB ARMv8.2 where descriptor bits [15:12] for bits [51:48] of output address Stage in bits[15:8]. Level in bits[7:0].

ptw_read_st1_invalid_long_descriptor

Page Table Walk (read). The descriptor that the PTW fetched is invalid. Fields:

desckind enum
: The kind of descriptor

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

input_address unsigned int
: Input address this lookup is for (after TBI normalisation).

ns bool
: Non-Secure

pa_address unsigned int
: Physical address.

ssd_ns enum
: Security state of stream

streamid unsigned int
: The StreamID this is for.

substreamid unsigned int
: The SubstreamID this is for.

trans_id unsigned int
: Transaction id

ttb_grain_stage_and_level unsigned int
: Which TTB used in bits[31:24]. Which Grain size used in bit[23:16]:- 0 -- 4 KiB 1 -- 16 KiB 2 -- 64 KiB 3 -- 64 KiB ARMv8.2 where descriptor bits [15:12] for bits [51:48] of output address Stage in bits[15:8]. Level in bits[7:0].

ptw_read_st1_leaf_long_descriptor

Page Table Walk (read). The descriptor that the PTW fetched is a block or page and this is the decode. Fields:

AF bool
: Access Flag.

AP21 enum
: The access permissions.

AttrIndx210 unsigned int
: The attribute index into the MAIR0/1.

DBM bool
: Dirty Bit Modifier. This only has meaning if running in AArch64 mode.

NS bool
: The encoding is for non-secure if this is a secure fetch.

PXN bool
: Privileged eXecute Never.

SH10 enum
: The shareability.

XN bool
: eXecute Never.

contiguous bool
: Contiguous hint.

desckind enum
: The kind of descriptor

hwu_pbha unsigned int
: Top four bits are appropriate CD.HWU*, bottom bits[62:59] of descriptor. Page Based Hardware Attributes: only valid on a bit-per-bit basis enabled by CD.HWU0*/CD.HWU1*.

input_address unsigned int
: Input address this lookup is for (after TBI normalisation).

nG bool
: not Global.

nT bool
: SMMUv3.2: If true then do not cache this entry in such a way that it can cause a TLB conflict abort (F_TLB_CONFLICT). The entry must produce a consistent result.

ns bool
: Non-Secure

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

output_address unsigned int
: Output address.

pa_address unsigned int
: Physical address.

ssd_ns enum
: Security state of stream

streamid unsigned int
: The StreamID this is for.

substreamid unsigned int
: The SubstreamID this is for.

trans_id unsigned int
: Transaction id

ttb_grain_stage_and_level unsigned int
: Which TTB used in bits[31:24]. Which Grain size used in bit[23:16]:- 0 -- 4 KiB 1 -- 16 KiB 2 -- 64 KiB 3 -- 64 KiB ARMv8.2 where descriptor bits [15:12] for bits [51:48] of output address Stage in bits[15:8]. Level in bits[7:0].

ptw_read_st1_table_long_descriptor

Page Table Walk (read). The descriptor that the PTW fetched is a Table, this decodes the fields. Fields:

APTable enum
: Remove permissions independently of subsequent descriptors.

NSTable bool
: The next level table descriptor is forced to non-secure.

PXNTable bool
: Force PXN independently of subsequent descriptors.

TableAddress unsigned int
: Address of the next table.

XNTable bool
: Force XN independently of subsequent descriptors.

desckind enum
: The kind of descriptor

input_address unsigned int
: Input address this lookup is for (after TBI normalisation).

ns bool
: Non-Secure

pa_address unsigned int
: Physical address.

ssd_ns enum
: Security state of stream

streamid unsigned int
: The StreamID this is for.

substreamid unsigned int
: The SubstreamID this is for.

trans_id unsigned int
: Transaction id

ttb_grain_stage_and_level unsigned int
: Which TTB used in bits[31:24]. Which Grain size used in bit[23:16]:- 0 -- 4 KiB 1 -- 16 KiB 2 -- 64 KiB 3 -- 64 KiB ARMv8.2 where descriptor bits [15:12] for bits [51:48] of output address Stage in bits[15:8]. Level in bits[7:0].

ptw_read_st2_invalid_descriptor

Page Table Walk (read). The descriptor that the PTW fetched is invalid. Fields:

desckind enum
: The kind of descriptor

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

input_address unsigned int
: Input address this lookup is for (after TBI normalisation).

ns bool
: Non-Secure

pa_address unsigned int
: Physical address.

ssd_ns enum
: Security state of stream

streamid unsigned int
: The StreamID this is for.

substreamid unsigned int
: The SubstreamID this is for.

trans_id unsigned int
: Transaction id

ttb_grain_stage_and_level unsigned int
: Which TTB used in bits[31:24]. Which Grain size used in bit[23:16]:- 0 -- 4 KiB 1 -- 16 KiB 2 -- 64 KiB 3 -- 64 KiB ARMv8.2 where descriptor bits [15:12] for bits [51:48] of output address Stage in bits[15:8]. Level in bits[7:0].

ptw_read_st2_leaf_descriptor

Page Table Walk (read). The descriptor that the PTW fetched is a block or page and this is the decode. Fields:

AF bool
: Access Flag.

DBM bool
: Dirty Bit Modifier. This only has meaning if running in AArch64 mode.

HAP21 enum
: The access permissions.

MemAttr3_0 enum
: The memory attributes.

SH10 enum
: The shareability.

XN enum
: eXecute Never.

contiguous bool
: Contiguous hint.

desckind enum
: The kind of descriptor

input_address unsigned int
: Input address this lookup is for (after TBI normalisation).

nT bool
: SMMUv3.2: If true then do not cache this entry in such a way that it can cause a TLB conflict abort (F_TLB_CONFLICT). The entry must produce a consistent result.

ns bool
: Non-Secure

output_address unsigned int
: Output address.

pa_address unsigned int
: Physical address.

s2hwu_pbha unsigned int
: Top four bits are STE.S2HWU*, bottom bits[62:59] of descriptor. Page Based Hardware Attributes: only valid on a bit-per-bit basis enabled by STE.S2HWU*.

ssd_ns enum
: Security state of stream

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

streamid unsigned int
: The StreamID this is for.

substreamid unsigned int
: The SubstreamID this is for.

trans_id unsigned int
: Transaction id

ttb_grain_stage_and_level unsigned int
: Which TTB used in bits[31:24]. Which Grain size used in bit[23:16]:- 0 -- 4 KiB 1 -- 16 KiB 2 -- 64 KiB 3 -- 64 KiB ARMv8.2 where descriptor bits [15:12] for bits [51:48] of output address Stage in bits[15:8]. Level in bits[7:0].

ptw_read_st2_table_descriptor

Page Table Walk (read). The descriptor that the PTW fetched is a Table, this decodes the fields. Fields:

APTable enum
: Remove permissions independently of subsequent descriptors.

PXNTable bool
: Force PXN independently of subsequent descriptors.

TableAddress unsigned int
: Address of the next table.

XNTable bool
: Force XN independently of subsequent descriptors.

desckind enum
: The kind of descriptor

input_address unsigned int
: Input address this lookup is for (after TBI normalisation).

ns bool
: Non-Secure

pa_address unsigned int
: Physical address.

ssd_ns enum
: Security state of stream

streamid unsigned int
: The StreamID this is for.

substreamid unsigned int
: The SubstreamID this is for.

trans_id unsigned int
: Transaction id

ttb_grain_stage_and_level unsigned int
: Which TTB used in bits[31:24]. Which Grain size used in bit[23:16]:- 0 -- 4 KiB 1 -- 16 KiB 2 -- 64 KiB 3 -- 64 KiB ARMv8.2 where descriptor bits [15:12] for bits [51:48] of output address Stage in bits[15:8]. Level in bits[7:0].

raw_register_end_read

The raw register read transaction. This is the transaction as directed to the register port. Fields:

id unsigned int
: Id of this transaction.

ok bool
: Was the access OK? The bus response will always be OK, but is the register access reasonable.

read_data unsigned int
: The data read.

raw_register_end_write

The raw register write transaction. This is the transaction as directed to the register port. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

id unsigned int
: Id of this transaction.

ok bool
: Was the access OK? The bus response will always be OK, but is the register access reasonable.

raw_register_start_read

The raw register read transaction. This is the transaction as directed to the register port. Fields:

address unsigned int
: Address of transaction.

id unsigned int
: Id of this transaction. Top bit is set if is a debug transaction.

ns enum
: Is the transaction non-secure?

number_of_beats unsigned int
: Number of beats.

size_of_beat_in_bytes unsigned int
: Size of each beat in bytes.

raw_register_start_write

The raw register write transaction. This is the transaction as directed to the register port. Fields:

address unsigned int
: Address of transaction

id unsigned int
: Id of this transaction. Top bit is set if is a debug transaction.

ns enum
: Is the transaction non-secure?

number_of_beats unsigned int
: Number of beats.

size_of_beat_in_bytes unsigned int
: Size of each beat in bytes.

write_data unsigned int
: The data to write.

register_disallowed_read_string

A text representation of the read of a register that was disallowed. Fields:

out string
: The text description of the register value read.

register_disallowed_write_string

A text representation of the write of a register write that was disallowed. Fields:

in string
: The text description of the register value written.

register_read_reserved

A text representation of an access to a register address that is reserved. Fields:

in string
: The text description of the register value.

register_read_string

A text representation of the read of a register. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

out string
The text description of the register value read.

register_write_reserved

A text representation of an access to a register address that is reserved or a write.to a RES0 field in a register. Fields:

in string
The text description of the register value.

register_write_string

A text representation of the write of a register. Fields:

in string
The text description of the register value written.

s_cmd_sync_completed_irq

Secure CMD_SYNC completed its IRQ action. Fields:

cmd_id unsigned int
The ID of the command being issued.
cons_incl_wrap unsigned int
CONS pointer of CMD_SYNC.

s_cmd_sync_completed_no_action

Secure CMD_SYNC completed, there was no CMD_SYNC action (SIG_NONE). Fields:

cmd_id unsigned int
The ID of the command being issued.
cons_incl_wrap unsigned int
CONS pointer of CMD_SYNC.

s_cmd_sync_completed_sev

Secure CMD_SYNC completed its action of SIG_SEV. Fields:

cmd_id unsigned int
The ID of the command being issued.
cons_incl_wrap unsigned int
CONS pointer of CMD_SYNC.

s_cmd_sync_completed_with_error

Secure CMD_SYNC completed, however, there was an error associated with the completion of the CMD_SYNC. Fields:

cerror enum
The error associated with the CMD_SYNC
cmd_id unsigned int
The ID of the command being issued.
cons_incl_wrap unsigned int
CONS pointer of CMD_SYNC.

s_cmd_sync_issued

Secure CMD_SYNC has now completed issuing and we now know if something is hazarding the CMD_SYNC from completing. Fields:

cmd_id unsigned int
The ID of the command being issued.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

cons_incl_wrap unsigned int
: CONS pointer of CMD_SYNC.

hazarded bool
: The CMD_SYNC is currently hazarded from completing.

s_cmd_sync_issuing

Secure CMD_SYNC is starting to issue, we are going to start accounting for which things have to be part of the prior set of things that have to complete before the CMD_SYNC can complete. Fields:

cmd_id unsigned int
: The ID of the command being issued.

cons_incl_wrap unsigned int
: CONS pointer of CMD_SYNC.

s_cmd_sync_starting_completion_action_irq

Secure CMD_SYNC starting IRQ completion action. Fields:

cmd_id unsigned int
: The ID of the command being issued.

cons_incl_wrap unsigned int
: CONS pointer of CMD_SYNC.

msiaddress unsigned int
: The MSI address to use (or 0 if not sending an MSI).

msidata unsigned int
: The MSI data to send.

s_cmd_sync_starting_completion_action_sev

Secure CMD_SYNC starting its completion action of SIG_SEV. Fields:

cmd_id unsigned int
: The ID of the command being issued.

cons_incl_wrap unsigned int
: CONS pointer of CMD_SYNC.

s_cmdq_cmd_sync_error

The CMD_SYNC can fault if an ATC Invalidate timeout occurs. The timeout will be attached to the next CMD_SYNC on the queue on which the invalidate was launched. Fields:

cerror enum
: The CERROR to attach.

cmd_queue unsigned int
: The top bits indicate the CMDQ id. These bits are attached to all cmd_ids generated for this queue.

s_cmdq_cmd_sync_error_delivered

The CMD_SYNC can fault if an ATC Invalidate timeout occurs. This is the CMD_SYNC delivering that fault. Fields:

cerror enum
: The CERROR to attach.

cmd_id unsigned int
: Command ID of the CMD_SYNC

s_cmdq_ecmdq_enack

Trace the changes to ENACK. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

cerror enum
: The error being exposed, if any.

cmd_queue unsigned int
: The top bits indicate the CMDQ id. These bits are attached to all cmd_ids generated for this queue.

enack bool
: The value of ENACK.

forced_error bool
: The queue was enabled with ERR != ERRACK and so the implementation chooses to trigger the error reporting mechanism.

s_cmdq_issue

A command is actually being executed. Fields:

cmd_id unsigned int
: Command ID of the command being executed.

cons unsigned int
: CONS of the command.

what string
: What is being executed.

s_cmdq_pointers_state

The command queue pointers. Fields:

cmd_queue unsigned int
: The top bits indicate the CMDQ id. These bits are attached to all cmd_ids generated for this queue.

disable_fetch bool
: The model has decided that as you programmed PROD/CONS inconsistently that it would not fetch as it has no confidence as to which are commands and which are not.

q_issuable_and_issued_cons unsigned int
: The CONS pointer of the region that is issuable and/or issued (including wrap).

q_issuable_and_issued_prod unsigned int
: The PROD pointer of the region that is issuable and/or issued (including wrap).

q_programmer_cons unsigned int
: The CONS pointer as seen by the programmer (including wrap).

q_programmer_prod unsigned int
: The PROD pointer as seen by the programmer (including wrap).

q_unfetched_cons unsigned int
: The CONS pointer of the region unfetched (including wrap).

q_unfetched_prod unsigned int
: The PROD pointer of the region unfetched (including wrap).

s_cmdq_state

A description of what the CMDQ can do now. Fields:

can_cerror bool
: There is a CERROR that can be recognised.

can_cmd_sync bool
: There is a CMD_SYNC that can be recognised.

can_fetch bool
: The SMMU can fetch

can_issue bool
: There are fetched commands that are eligible to be issued.

can_update_cons bool
: The programmer view of CMDQ_CONS can be moved to say some have been consumed.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

cmd_queue unsigned int
: The top bits indicate the CMDQ id. These bits are attached to all cmd_ids generated for this queue.

cmd_sync enum
: Why a CMD_SYNC cannot be issued

cmdqen bool
: CMDQEN, when enabled then the CMDQ can fetch commands

cmdqen_1_to_0_unacknowledged bool
: CMDQEN has been set to 0, but we have yet to acknowledge it.

current-real-cerror enum
: The current CERROR that would be visible to the programmer if there was an error.

fetch-disabled_due_to_programmer_error bool
: The PROD/CONS pointer was inconsistent and the model decided to stop processing commands.

fetch-number_of_unfetched_commands unsigned int
: The number of commands that have yet to be fetched from memory, or ~0u if CMDQEN == 0.

fetch-state enum
: The state of the fetch state machine

fetch-too_many_commands_outstanding_so_not_fetching_any_more bool
: If there are too may issued commands, or pending to be issued then we won't fetch any more.

issue-number_of_issuable_commands unsigned int
: The number of commands that have been fetched but not yet issued.

issue-number_of_unissuable_commands unsigned int
: This might include commands behind an illegal command, or behind a CMD_SYNC. Under some circumstances this can include a command representing an external abort.

issue-state enum
: The internal issued state machine state

outstanding_fetches_since_reset unsigned int
: The number of outstanding fetches.

s_eventq_adding_to_cmd_sync_prior_set

The event created by the specified transaction id must become visible to the programmer before the CMD_SYNC can complete. Fields:

trans_id unsigned int
: The transaction id that caused the event.

s_eventq_auto_retry_stalled_transaction

A transaction that was stalled but not yet reported on the Event Queue was retried. The retry can happen if the transaction didn't write because it was de-duplicated behind a reported stalling transaction and that stalling transaction was CMD_RESUMEd, or if the queue was unwriteable and the queue is now writeable. Fields:

trans_id unsigned int
: Transaction id of the auto-retried transaction.

why enum
: Why the transaction retried.

s_eventq_cmd_sync_unhazarded

The CMD_SYNC has been unhazarded as the prior set is empty and as much of it as required is visible in the programmers' view. Fields:

prod_incl_wrap unsigned int
: The prod index including wrap bit that was required to become visible.

prog_prod_incl_wrap unsigned int
: The current programmer-visible prod including wrap bit.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

s_eventq_eventqen_and_eventqenack

Trace the values of EVENTQEN and EVENTQENACK as they change. Fields:

about enum
: Some more information about the queue state.

cons_incl_wrap unsigned int
: The CONS pointer (including wrap bits).

eventqen bool
: EVENTQEN.

eventqenack bool
: EVENTQENACK.

prod_incl_wrap unsigned int
: The PROD pointer (including wrap bits).

s_eventq_external_abort

A particular eventq record got an external abort. The event will subsequently appear in the ns/ s_eventq_lost_event_records trace. Fields:

prod unsigned int
: The prod pointer (excluding wrap bit) of the event record that aborted.

trans_id unsigned int
: The transaction id that produced the event record that aborted.

s_eventq_lost_event_records

Event records that are lost and never record appear on this trace. Fields:

dword0 unsigned int
: Double word 0 of event.

dword1 unsigned int
: Double word 1 of event.

dword2 unsigned int
: Double word 2 of event.

dword3 unsigned int
: Double word 3 of event.

event enum
: Event code that was thrown away.

prod unsigned int
: The PROD index of the event if appropriate or ~0u otherwise.

reason enum
: The reason the record was lost.

trans_id unsigned int
: The transaction id that this corresponds to.

s_eventq_overflow_acknowledged

When events are lost then they appear on the eventq_lost_event_records_* trace. This will activate the overflow flag in the corresponding SMMU_s_EVENTQ_PROD register. Software can then acknowledge that flag using the overflow acknowledge flag in SMMU_s_EVENTQ_CONS. Fields:

bad bool
: If true then the programmer has attempted to acknowledge an overflow condition that didn't exist.

count_of_events_lost_due_to_overflow unsigned int
: This is the number of events that were lost because the event queue was full. This does not include any lost because of an EVENTQ_ABT_ERR.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

s_eventq_pending_event_records

Event records that are pending to be dealt with by the eventq state machine. A specific event can be traced multiple times if it becomes part of the prior set of transactions of a CMD_SYNC that has to complete before the CMD_SYNC is allowed to be completed. Fields:

dword0 unsigned int
: Double word 0 of event.

dword1 unsigned int
: Double word 1 of event.

dword2 unsigned int
: Double word 2 of event.

dword3 unsigned int
: Double word 3 of event.

event enum
: Event code.

is_prior_set bool
: The transaction forms part of the prior set of a CMD_SYNC that have to become visible (or thrown away if necessary) before the CMD_SYNC can complete

trans_id unsigned int
: The transaction id that this corresponds to.

s_eventq_prod_cons_updated

This records when the programmer visible PROD/CONS pointer is updated. More can be visible in memory than are advertised in the programmer-visible PROD pointer. This also traces the latest PROD that could have been advertised. Fields:

mem_prod unsigned int
: The PROD pointer value for all records that are visible in the memory.

num_events_available unsigned int
: The number of events available to the programmer.

prog_cons unsigned int
: The programmer-visible CONS pointer.

prog_prod unsigned int
: The programmer-visible PROD pointer.

what enum
: What changed?

s_eventq_raising_eventq_abt_err

EVENTQ_ABT_ERR can now be raised.

s_eventq_removing_from_cmd_sync_prior_set

The event captured by a CMD_SYNC and created by the specified transaction id is now in memory. Fields:

prod unsigned int
: The prod index (no wrap bit) that must become visible to the programmer to see this record. If not head_of_line then the record will not become visible and this field is meaningless.

reason enum
: The reason the event was removed.

trans_id unsigned int
: The transaction id that caused the event.

s_eventq_stashing_unreported_stalled_transaction_for_auto_retry

A stall event record would have been lost, instead it is stashed away to auto-retry when it can. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

dword0 unsigned int
: Double word 0 of event.

dword1 unsigned int
: Double word 1 of event.

dword2 unsigned int
: Double word 2 of event.

dword3 unsigned int
: Double word 3 of event.

event enum
: Event code that was thrown away.

reason enum
: The reason the record was lost.

trans_id unsigned int
: The transaction id that this corresponds to.

s_eventq_write_event

An event is committed to being written. Fields:

dword0 unsigned int
: Double word 0 of event.

dword1 unsigned int
: Double word 1 of event.

dword2 unsigned int
: Double word 2 of event.

dword3 unsigned int
: Double word 3 of event.

event enum
: Event code that was thrown away.

prod unsigned int
: The PROD pointer (without wrap) that this event will be written to.

stall_event bool
: Describes an event for a stalled transaction.

trans_id unsigned int
: Transaction ID that generated this event.

s_eventq_write_event_visible_in_memory

An event is now visible in memory. Fields:

abort bool
: The write aborted.

dword0 unsigned int
: Double word 0 of event.

dword1 unsigned int
: Double word 1 of event.

dword2 unsigned int
: Double word 2 of event.

dword3 unsigned int
: Double word 3 of event.

event enum
: Event code that was thrown away.

prod unsigned int
: The PROD pointer (without wrap) that this event will be written to.

stall_event bool
: Describes an event for a stalled transaction.

trans_id unsigned int
: Transaction ID that generated this event.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

s_gerror_inverted

A GERROR bit was attempted to be inverted to record a fault. Fields:

already_different_to_gerrorn bool
: True if the bit was already different to the corresponding bit in GERRORN and so it was not inverted.

gerrorn unsigned int
: GERRORN register value.

interrupt_potentially_sent bool
: An interrupt is potentially sent, depending on IRQEN.
GERROR(N).MSI_GERROR_ABT_ERR does not send an interupt.

new_gerror unsigned int
: GERROR register value after any invertion.

which enum
: Which bit was attempted to be inverted.

s_gerrorn_acknowledge

A GERROR was acknowledged by SW. Fields:

which enum
: Which GERROR was acknowledged.

sec_override

Controls non-secure accesses to some registers. Fields:

value bool
: The value of the signal.

sev

Send a SEV. Fields:

why enum
: Why the SEV was generated.

smmu_atc_inv

The CMD_ATC_INV command is sent. Fields:

address unsigned int
: Untranslated Address to invalidate.

cmd_id unsigned int
: Command id, if top-bit is set then was issued from the Non-secure CMDQ.

cons unsigned int
: CONS of the command.

global bool
: Global flag.

log2_size_in_bytes unsigned int
: Log 2 of the size in bytes of the region to invalidate.

ns_number_in_flight unsigned int
: Number of ATC invalidates in flight issued from the Non-secure CMDQ.

ns_number_waiting_to_go_out unsigned int
: The total number of ATC invalidates waiting to be pushed out from the Non-secure CMDQ.

s_number_in_flight unsigned int
: Number of ATC invalidates in flight issued from the Secure CMDQ.

s_number_waiting_to_go_out unsigned int
: The total number of ATC invalidates waiting to be pushed out from the Secure CMDQ.

streamid unsigned int
: StreamID in the invalidate command.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

substreamid unsigned int
: SubstreamID or ~0u if SSValid == false.

smmu_atc_inv_completed

The CMD_ATC_INV command completed. Fields:

cmd_id unsigned int
: Command id, if top-bit is set then was issued from the Non-secure CMDQ.

ns_number_in_flight unsigned int
: Number of ATC invalidates in flight now this one has completed, that were issued by the Non-secure CMDQ.

ns_number_waiting_to_go_out unsigned int
: The total number of ATC invalidates waiting to be pushed out from the Non-secure CMDQ.

response enum
: The response.

s_number_in_flight unsigned int
: Number of ATC invalidates in flight now this one has completed, that were issued by the Secure CMDQ.

s_number_waiting_to_go_out unsigned int
: The total number of ATC invalidates waiting to be pushed out from the Secure CMDQ.

smmu_atc_inv_end

An ATC invalidate request has been started. Fields:

PASID_global bool
: If has a SubstreamID (PASID) then if it is 'global' or not.

address unsigned int
: The Untranslated Address to use.

log2_size_in_bytes unsigned int
: Log2 of the size of the region in bytes to invalidate.

response enum
: The response to the ATC invalidate

streamid unsigned int
: The StreamID to use.

substreamid unsigned int
: The SubstreamID to use (or ~0u if not used). This corresponds to the PASID.

smmu_atc_inv_start

An ATC invalidate request has been started. Fields:

PASID_global bool
: If has a SubstreamID (PASID) then if it is 'global' or not.

address unsigned int
: The Untranslated Address to use.

log2_size_in_bytes unsigned int
: Log2 of the size of the region in bytes to invalidate.

streamid unsigned int
: The StreamID to use.

substreamid unsigned int
: The SubstreamID to use (or ~0u if not used). This corresponds to the PASID.

smmu_ats_initial

This is the initial ATS request. Fields:

ia unsigned int
: Input address

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

max_number_of_replies unsigned int
: The maximum number of replies allowed to return

no_write bool
: The NW (no write flag) of the ATS request. If clear then the requester is going to do a write.

pasid_execute_requested bool
: If there is a PASID (substreamid != ~0u) then this represents the requester asking for execution permissions.

pasid_privileged_mode_requested bool
: If there is a PASID (substreamid != ~0u) then this represents the requester asking for the privileged mode's permissions.

ssd_ns bool
: Incoming SSD, only non-secure ATS requests are legal.

streamid unsigned int
: StreamID of the ATS request.

substreamid unsigned int
: SubstreamID (which is identical to the PASID) of the ATS request. If no PASID-prefix is sent then this is ~0u

tbu unsigned int
: Translation Buffer Unit number

smmu_ats_reply_failure

This is an ATS reply indicating failure. Fields:

event enum
: Equivalent event number that would have been generated for an equivalent ordinary transaction.

failure enum
: What is the failure response code?

state enum
: The transaction state of the successfully ATS request.

smmu_ats_reply_success

This is an ATS reply, typically the SMMU will only return a single response, even if the requester indicated it could accept more replies. NOTE that the SMMU responds with 'success' in some cases when a fault is encountered and RW==0. Fields:

N bool
: Non-snooped access. If one then the requester must clear the NoSnoop bit on transactions, unless otherwise enabled in a Function-specific manner.

P bool
: Privileged mode. These permissions related to privileged mode.

RWX enum
: Read/Write/Execute

U bool
: Untranslated access. If one, and RW !=0 then use UntranslatedAccesses for the allowed accesses by RW(X).

inner enum
: The inner cacheability attributes to use for TranslatedAccesses

input_address unsigned int
: Input address of the ATS request

instcfg enum
: The STE.INSTCFG field

outer enum
: The outer cacheability attributes to use for TranslatedAccesses

privcfg enum
: The STE.PRIVCFG field

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

shareability unsigned int
: The shareability to use for TranslatedAccesses

size unsigned int
: The size of the region covered by this translation.

state enum
: The transaction state of the successfully ATS request.

translated_address unsigned int
: If RW!=0 && U != 0, then the Translated Address that a TranslatedAccess can be made with

smmu_final_transaction

This is the transaction group request to remap has completed one set of remapping. For ordinary transactions, this represents a bundle of transactions with the same attributes but different addresses within a certain range around the address. If it stalls then it will report through this trace source, stall (stag_if_stalling != ~0u) and when resume will issue another smmu_initial_transaction as it undergoes remapping again. This trace source can also represent part of the process for ATOS/ATS or finding STE.PPAR for PRI requests that need to be auto-responded to. Fields:

axmmuflow enum
: The AxMMUFLOW for this transaction group. stallable -- allow the transaction to be stalled if configured. translated-access -- is a PCIe Translated Access non-stallable -- do not stall transaction even if configured. transfault-flow -- do not report for Translation Related Faults, but tell client device to use a page request mechanism instead (for example PRI).

cmo_point enum
: The point associated with the CMO, if applicable

input_address unsigned int
: The input address of the transaction group.

input_inner enum
: Inner cacheability for the input attributes.

input_outer enum
: Outer cacheability for the input attributes.

input_prot enum
: The instruction/data, bus-ns/s and privileged/user marking

input_sh enum
: Shareability for the input attributes.

ipa_address unsigned int
: The IPA of the transaction.

kind enum
: The kind of access this transaction group that this represents.

mpam_pmg_and_partid unsigned int
: Bits[23:16] are the MPAM PMG. Bits[15:0] are the physical PARTID.

operation enum
: The kind of operation that this represents.

output_address unsigned int
: The input address of the transaction group.

output_inner enum
: Inner cacheability for the output attributes.

output_outer enum
: Outer cacheability for the output attributes.

output_prot enum
: The instruction/data, bus-ns/s and privileged/user marking

output_sh enum
: Shareability for the output attributes.

output_vmid unsigned int
: The output VMID/GBPA.IMPDEF or ~0u if not valid.

size_of_region_in_bytes unsigned int
: An imp def size of region for which this translation is valid for.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ssd_ns enum
: The security state of the transaction

stag_if_stalling unsigned int
: This is the STAG used by the transaction if it is going to stall. It is ~0u if it is not going to stall.

state enum
: The final transaction state.

streamid unsigned int
: The StreamID of the transction.

substreamid unsigned int
: The SubstreamID of the transaction or 0xFFFFffff if no SubstreamID.

tbu unsigned int
: Translation Buffer Unit number.

trans_id unsigned int
: Transaction id. Top bit set if debug transaction.

smmu_initial_transaction

This is the transaction group request to remap is going to start one set of remapping. For ordinary transactions, this represents a bundle of transactions with the same attributes but different addresses within a certain range around the address. A stalling transaction will report through this trace source when it unstalls. This trace source can also represent part of the process for ATOS/ATS or finding STE.PPAR for PRI requests that need to be auto-responded to. Fields:

axmmuflow enum
: The AxMMUFLOW for this transaction group. stallable -- allow the transaction to be stalled if configured. translated-access -- is a PCIe Translated Access non-stallable -- do not stall transaction even if configured. transfault-flow -- do not report for Translation Related Faults, but tell client device to use a page request mechanism instead (for example PRI).

cmo_point enum
: The point associated with the CMO, if applicable

input_address unsigned int
: The input address of the transaction group.

input_inner enum
: Inner cacheability for the input attributes.

input_outer enum
: Outer cacheability for the input attributes.

input_prot enum
: The instruction/data, bus-ns/s and privileged/user marking

input_sh enum
: Shareability for the input attributes.

kind enum
: The kind of access this transaction group that this represents.

operation enum
: The kind of operation that this represents.

ssd_ns enum
: The security state of the transaction

streamid unsigned int
: The StreamID of the transction.

substreamid unsigned int
: The SubstreamID of the transaction or 0xFFFFffff if no SubstreamID.

tbu unsigned int
: Translation Buffer Unit number.

trans_id unsigned int
: Transaction id. Top bit set if debug transaction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

smmu_pmusnapshot_ack

Acknowledge the pmusnapshot_req to indicate the snapshot has occurred. This is part of a four-phase handshake. Fields:

pin_index unsigned int
: The pin index of the signal, or ~0u if not relevant.

pmcg_index enum
: The PMCG index.

value bool
: The value of the signal.

smmu_pmusnapshot_req

Take a snap shot of the PMU values as though SMMU_PMCG_CAPR.CAPTURE had been written. This is part of a four-phase handshake. Fields:

pin_index unsigned int
: The pin index of the signal, or ~0u if not relevant.

pmcg_index enum
: The PMCG index.

value bool
: The value of the signal.

smmu_poison_tw_data

Poison data has been returned to a table walk transaction. Fields:

bitmap_of_poison unsigned int
: The bitmap of which beats of the transaction where poisoned.

is_cas bool
: True if this is a compare-and-swap operation.

ns bool
: Non-secure state of the bus transaction.

number_of_64bit_beats unsigned int
: Number of beats this transaction fetched.

paddress unsigned int
: Physical address of the table walk transaction.

ras_group_id unsigned int
: If non-~0u then is the RAS group that we wish to attach the error to. Otherwise will use the IMP DEF policy.

ras_record_index_in_group unsigned int
: If non-~0u then is the RAS record index in the group that we wish to attach the error to. Otherwise will use the IMP DEF policy.

what enum
: What table walk was being performed.

smmu_pri_resp

The CMD_PRI_RESP command is queued to be sent to the PCIe system. Fields:

cmd_id unsigned int
: Command id

cons unsigned int
: CONS of the command.

prgindex unsigned int
: PRG Index.

response enum
: If has a SubstreamID (PASID) then if it is 'global' or not.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

streamid unsigned int
: StreamID in the invalidate command.

substreamid unsigned int
: SubstreamID or ~0u if SSValid == false.

smmu_pri_resp_nop

The CMD_PRI_RESP command was NOPped. Fields:

cmd_id unsigned int
: Command id

cons unsigned int
: CONS of the command.

prgindex unsigned int
: PRG Index.

response enum
: If has a SubstreamID (PASID) then if it is 'global' or not.

streamid unsigned int
: StreamID in the invalidate command.

substreamid unsigned int
: SubstreamID or ~0u if SSValid == false.

why enum
: Why the CMD_PRI_RESP was NOPped.

smmu_priq_resp_fake_return

A PRIQ Response is posted to the PCIe subsystem and so has no acknowledgement that it is received. However, in the model then we artificially know when the the PRIQ Response has been delivered to the PCIe subsystem, even if the ATC has not yet acted on it. Fields:

prgindex unsigned int
: The PRG Index this request corresponds to.

streamid unsigned int
: The StreamID to use.

substreamid unsigned int
: The SubstreamID to use (or ~0u if not used). This corresponds to the PASID.

what enum
: What happened.

smmu_priq_resp_start

A PRIQ Response has been posted to the PCIe subsystem. As the response is posted then there is no way of knowing when it is received by the EndPoint. Fields:

prgindex unsigned int
: The PRG Index this request corresponds to.

response enum
: If has a SubstreamID (PASID) then if it is 'global' or not.

streamid unsigned int
: The StreamID to use.

substreamid unsigned int
: The SubstreamID to use (or ~0u if not used). This corresponds to the PASID.

smmu_thread_wait_wake

Traces a thread's wait/wake status. Fields:

current_ticks unsigned int
: The current tick count of simulated time.

event enum
: What is happening to this thread.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

thread_index unsigned int
: The ID of this thread.

ticks unsigned int
: If the event relates to a time then this is held in this field. Otherwise, 0.

stall_transaction

A transaction is about to stall. Fields:

stag unsigned int
: STAG.

streamid unsigned int
: StreamID.

streamid_ns bool
: The StreamID was for the non-secure world.

trans_id unsigned int
: Transaction id.

stall_transaction_inhibited_by_STALL_MAX

A transaction is about to stall but the maximum number of transactions have stalled and we can't report this one to the event queue (even if non-full). Fields:

streamid unsigned int
: StreamID.

streamid_ns bool
: The StreamID was for the non-secure world.

trans_id unsigned int
: Transaction id.

stall_transaction_resuming

A stalled transaction is resuming. Fields:

stag unsigned int
: STAG if appropriate, or if was inhibited by STALL_MAX then 0xFAFA.

stallresult enum
: What the transaction resumed to do.

streamid unsigned int
: StreamID.

streamid_ns bool
: The StreamID was for the non-secure world.

trans_id unsigned int
: Transaction id.

start_ptw_read

Page Table Walk (read). This is the start of the physical access that the SMMU is making. Fields:

adomain enum
: The actual attributes of the access that was used, after IMP DEF mangling.

aprot enum
: The actual attributes of the access that was used, after IMP DEF mangling.

desckind enum
: The kind of descriptor

inner_cache enum
: The actual attributes of the access that was used, after IMP DEF mangling.

input_address unsigned int
: Input address this lookup is for (after TBI normalisation).

mpam_pmg_and_partid unsigned int
: Bits[23:16] are the MPAM PMG. Bits[15:0] are the physical PARTID.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ns bool
: Non-Secure

outer_cache enum
: The actual attributes of the access that was used, after IMP DEF mangling.

pa_address unsigned int
: Physical address.

ssd_ns enum
: Security state of stream

streamid unsigned int
: The StreamID this is for.

substreamid unsigned int
: The SubstreamID this is for.

trans_id unsigned int
: Transaction id

ttb_grain_stage_and_level unsigned int
: Which TTB used in bits[31:24]. Which Grain size used in bit[23:16]:- 0 -- 4 KiB 1 -- 16 KiB 2 -- 64 KiB 3 -- 64 KiB ARMv8.2 where descriptor bits [15:12] for bits [51:48] of output address Stage in bits[15:8]. Level in bits[7:0].

ste_cc.STE_commentary

A commentary from the cache about what is going on. Fields:

output string
: The stream output.

ste_cc.STE_entry_invalidated

An entry in the cache has been invalidated. Fields:

entry_id unsigned int
: The entry id that is being invalidated.

reason enum
: Why the entry is being invalidated.

ste_entry_allocated

An STE entry has been allocated. Fields:

entry_id unsigned int
: The id of the cache entry. Top bit is set if is a debug entry.

ns enum
: For the non-secure world

ste string
: A textual description of the STE.

streamid unsigned int
: The StreamID of the entry this will match.

sup_btm

The system supports BTM. Fields:

value bool
: The value of the signal.

sup_cohacc

The system supports cohacc. Fields:

value bool
: The value of the signal.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

sup_sev

The system supports SEV. Fields:

value bool
: The value of the signal.

tbu0_reset_in

The reset signal of the TBU0. Fields:

value bool
: The value of the signal.

tcu_reset_in

The reset signal. Fields:

value bool
: The value of the signal.

tlb_entry_allocated

A TLB entry has been allocated. Fields:

aset unsigned int
: The ASET of this entry.

asid unsigned int
: ASID if appropriate.

index unsigned int
: Index of TLB entry.

input_end_incl_address unsigned int
: The end inclusive address of the input range that this matches.

input_start_address unsigned int
: The start address of the input range that this matches.

output_end_incl_address unsigned int
: The end inclusive address of the output range.

output_start_address unsigned int
: The start address of the output range.

scheme enum
: The tagging scheme used

ssd_ns enum
: Security State

tbi bool
: Was the entry formed using Top Byte Ignore (TBI).

vmid unsigned int
: VMID if appropriate.

tlb_info_tlb_entries_overlap

A TLB entry was inserted into the TLB and it overlaps an existing entry. This isn't a problem as it was inserted in such a way that it architecturally works. Fields:

end_address_of_new_entry unsigned int
: End address of new entry.

end_address_of_old_entry unsigned int
: End address of old entry.

how_inserted enum
: How the entry was inserted.

index_of_new_entry unsigned int
: Index of new entry.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

index_of_old_entry unsigned int
: Index of old entry.

start_address_of_new_entry unsigned int
: Start address of new entry.

start_address_of_old_entry unsigned int
: Start address of old entry.

verbose_commentary

This is a verbose commentary on the translation process the SMMU is performing. Fields:

output string
: The stream output.

warning_MSI_EVENTQ_address_out_of_range_of_oas

The MSI address is configured outside of the range of the downstream address bus size. Fields:

address unsigned int
: The untruncated address of the MSI.

warning_MSI_GERROR_address_out_of_range_of_oas

The MSI address is configured outside of the range of the downstream address bus size. Fields:

address unsigned int
: The untruncated address of the MSI.

warning_MSI_PRIQ_address_out_of_range_of_oas

The MSI address is configured outside of the range of the downstream address bus size. Fields:

address unsigned int
: The untruncated address of the MSI.

warning_MSI_S_EVENTQ_address_out_of_range_of_oas

The MSI address is configured outside of the range of the downstream address bus size. Fields:

address unsigned int
: The untruncated address of the MSI.

warning_MSI_S_GERROR_address_out_of_range_of_oas

The MSI address is configured outside of the range of the downstream address bus size. Fields:

address unsigned int
: The untruncated address of the MSI.

warning_MSI_pmcg_address_out_of_range_of_oas

The MSI Address of the Performance Monitor Counter Group (PMCG) is out of range of the OAS and so will be silently truncated. Fields:

address unsigned int
: The untruncated address of the MSI.

pmcg_index unsigned int
: Index of the PMCG.

ssd_ns bool
: True if this PMCG controlled by the non-secure world.

warning_discarding_interrupt_EVENTQ_as_irqen_low

Interrupt generation is turned off by SMMU_IRQ_CTRL.EVENTQ_IRQEN.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

warning_discarding_interrupt_GERROR_as_irqen_low

Interrupt generation is turned off by SMMU_IRQ_CTRL.GERROR_IRQEN.

warning_discarding_interrupt_PRIQ_as_irqen_low

Interrupt generation is turned off by SMMU_S_IRQ_CTRL.PRIQ_IRQEN.

warning_discarding_interrupt_S_EVENTQ_as_irqen_low

Interrupt generation is turned off by SMMU_S_IRQ_CTRL.EVENTQ_IRQEN.

warning_discarding_interrupt_S_GERROR_as_irqen_low

Interrupt generation is turned off by SMMU_S_IRQ_CTRL.GERROR_IRQEN.

warning_gerrorn_bad_acknowledge

A GERROR was acknowledged by SW in the GERRORN register that did not have an active error. The result is UNPREDICTABLE if this will generate an interrupt or not. Fields:

which enum
: Which GERROR was acknowledge when there was no active error.

warning_reg_after_doesnt_match_written_value

A write occurred that tried to set bits in a register, that for one reason or another, failed to get written. Fields:

desc string
: The textual description of what happened.

warning_s_gerrorn_bad_acknowledge

A GERROR was acknowledged by SW in the GERRORN register that did not have an active error. The result is UNPREDICTABLE if this will generate an interrupt or not. Fields:

which enum
: Which GERROR was acknowledge when there was no active error.

what_going_to_do_with_terminated_event

A terminating transaction has produced an event, this tells you what the model is going to do with the event. Fields:

CD.S bool
: The CD.S field if available.

S2 bool
: The event is related to Stage 2.

STE.S1STALLD bool
: The STE.S1STALLD field if available.

STE.S2S bool
: The STE.S2S field if available.

aborts bool
: The transaction will abort.

axmmuflow enum
: The AxMMUFLOW for this transaction group. stallable -- allow the transaction to be stalled if configured. translated-access -- is a PCIe Translated Access non-stallable -- do not stall transaction even if configured. transfault-flow -- do not report for Translation Related Faults, but tell client device to use a page request mechanism instead (for example PRI).

is_tr_fault bool
: Is a Translation Related fault.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

reports bool
: The transaction will attempt to report.

ssd_ns bool
: The transaction is classified as SSD non-secure.

supports_stall_model bool
: The implementation supports the stall model.

trans_id unsigned int
: The transaction id.

why_abort_decision enum
: The reason why the transaction aborted/did not abort.

why_report_decision enum
: The reason why the transaction reported/did not report.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.85 MMU_700 - trace

This section describes the trace sources.

ArchMsg.Error.error

These messages are about activity occurring on the SMMU that is considered an error. Messages will only come out here if parameter all_error_messages_through_trace is true. Fields:

output string
: The stream output

ArchMsg.Error.fetch_from_memory_type_not_supporting_httu

A descriptor fetch from an HTTU-enabled translation regime to an unsupported memory type was made. Whilst the fetch itself may succeed, if an update to the descriptor was attempted then it would fail. Fields:

address unsigned int
: The address of the descriptor fetch.

desc_inner enum
: Inner cacheability of descriptor

desc_outer enum
: Outer cacheability of descriptor

desc_prot enum
: The instruction/data, bus-ns/s and privileged/user marking

desc_sh enum
: Shareability of descriptor

ssd_ns bool
: Is the Security State Determination of the transaction non-secure.

stage unsigned int
: The stage at which we had a problem.

streamid unsigned int
: The StreamID of the transaction

substreamid unsigned int
: The SubstreamID of the transaction (or ~0u if no substreamid)

trans_id unsigned int
: Transaction id.

ArchMsg.Error.ns_cmdq

There is an error associated with the command queue. Fields:

cmd_id unsigned int
: Command ID of the command in error.

cons unsigned int
: CONS of the command.

what string
: What is wrong.

ArchMsg.Error.priq_streamid_truncated

The SMMU received a PCIe PRI request with a StreamID) that was larger than that which the SMMU has been configured for. The StreamID that appears in the PRIQ entry will be truncated. Fields:

actual_streamid unsigned int
: The actual StreamID that this request has.

sidsize unsigned int
: The bit width of the SMMU for StreamIDs, as indicated by SMMU_IDR1.SIDSIZE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

trans_id unsigned int
: The transaction ID that identifies this request.

trunc_streamid unsigned int
: The truncated StreamID that will appear in the PRIQ entry.

ArchMsg.Error.priq_substreamid_truncated

The SMMU received a PCIe PRI request with a PASID prefix (SubstreamID) that was larger than that which the SMMU has been configured for. The SubstreamID that appears in the PRIQ entry will be truncated. Fields:

actual_substreamid unsigned int
: The actual SubstreamID that this request has.

ssidsize unsigned int
: The bit width of the SMMU for SubstreamIDs, as indicated by SMMU_IDR1.SSIDSIZE.

trans_id unsigned int
: The transaction ID that identifies this request.

trunc_substreamid unsigned int
: The truncated SubstreamID that will appear in the PRIQ entry.

ArchMsg.Error.s_cmdq

There is an error associated with the command queue. Fields:

cmd_id unsigned int
: Command ID of the command in error.

cons unsigned int
: CONS of the command.

what string
: What is wrong.

ArchMsg.Error.tlb_entries_overlap

A TLB entry was attempted to be inserted into the TLB and was determined that it overlaps an existing entry. This check is not perfect but will catch simple errors. Fields:

do_f_tlb_conflict bool
: Chosen to perform an F_TLB_CONFLICT.

end_address_of_new_entry unsigned int
: End address of new entry.

end_address_of_old_entry unsigned int
: End address of old entry.

index_of_new_entry unsigned int
: Index of new entry.

index_of_old_entry unsigned int
: Index of old entry.

start_address_of_new_entry unsigned int
: Start address of new entry.

start_address_of_old_entry unsigned int
: Start address of old entry.

why enum
: Why it is an error for these two entries to overlap.

ArchMsg.Error.tlb_entry_not_invalidated_due_to_ril

An entry in the cache was not invalidated even though in the right address range because of the RIL fields of the command the entry do not match. Fields:

cmd_num unsigned int
: The NUM field of the RIL part of the command.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

cmd_ril_tg enum
: The RIL_TG field of the RIL part of the command.

cmd_ril_ttl unsigned int
: The TTL field of the RIL part of the command, zero means any level.

cmd_scale unsigned int
: The SCALE field of the RIL part of the command.

entry_id unsigned int
: The entry id that is being invalidated.

tlb_entry string
: The TLB entry.

ArchMsg.Error.vatos_sel_vmid_out_of_range

The SMMU_(S_)VATOS_SEL.VMID field was programmed with a VMID that was too wide for this implementation (SMMU_IDR0.VMID16 == 0). Fields:

ssd_ns bool
: The security state of the VATOS interface

vmid unsigned int
: The VMID programmed.

ArchMsg.Info.info

These are information messages about what is happening in the SMMU. Fields:

output string
: The stream output

ArchMsg.Warning.CMD_RESUME_no_transaction_resumed

A CMD_RESUME was issued that matched no transaction. Fields:

stag unsigned int
: STAG in the CMD_RESUME.

streamid unsigned int
: StreamID in the CMD_RESUME.

streamid_ns bool
: The StreamID was for the non-secure world.

ArchMsg.Warning.atc_inv_strange

Something was odd about the CMD_ATC_INV. Fields:

cmd_id unsigned int
: Command id

why string
: Why the CMD_ATC_INV was strange

ArchMsg.Warning.bad_conf_reset_of_SMMU_GBPA_ABORT

The signal was not driven at reset by the external system, but was driven after we first started using the default value we had assumed. The external system, if it is going to drive the signal, must do so at first reset so that both sides of the connection can agree on the value of this signal. Fields:

default_value bool
: The default value of the signal we have been assuming.

value bool
: The value of the signal.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.bad_conf_reset_of_SMMU_S_GBPA_ABORT

The signal was not driven at reset by the external system, but was driven after we first started using the default value we had assumed. The external system, if it is going to drive the signal, must do so at first reset so that both sides of the connection can agree on the value of this signal. Fields:

default_value bool
: The default value of the signal we have been assuming.

value bool
: The value of the signal.

ArchMsg.Warning.bad_sec_override

The signal was not driven at reset by the external system, but was driven after we first started using the default value we had assumed. The external system, if it is going to drive the signal, must do so at first reset so that both sides of the connection can agree on the value of this signal. Fields:

default_value bool
: The default value of the signal we have been assuming.

value bool
: The value of the signal.

ArchMsg.Warning.bad_sup_btm

The signal was not driven at reset by the external system, but was driven after we first started using the default value we had assumed. The external system, if it is going to drive the signal, must do so at first reset so that both sides of the connection can agree on the value of this signal. Fields:

default_value bool
: The default value of the signal we have been assuming.

value bool
: The value of the signal.

ArchMsg.Warning.bad_sup_cohacc

The signal was not driven at reset by the external system, but was driven after we first started using the default value we had assumed. The external system, if it is going to drive the signal, must do so at first reset so that both sides of the connection can agree on the value of this signal. Fields:

default_value bool
: The default value of the signal we have been assuming.

value bool
: The value of the signal.

ArchMsg.Warning.bad_sup_httu

The signal was not driven at reset by the external system, but was driven after we first started using the default value we had assumed. The external system, if it is going to drive the signal, must do so at first reset so that both sides of the connection can agree on the value of this signal. Fields:

default_value bool
: The default value of the signal we have been assuming.

value bool
: The value of the signal.

ArchMsg.Warning.bad_sup_sev

The signal was not driven at reset by the external system, but was driven after we first started using the default value we had assumed. The external system, if it is going to drive the signal, must do so at first reset so that both sides of the connection can agree on the value of this signal. Fields:

default_value bool
: The default value of the signal we have been assuming.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

value bool
The value of the signal.

ArchMsg.Warning.bad_tbu0_reset_in

The signal was not driven at reset by the external system, but was driven after we first started using the default value we had assumed. The external system, if it is going to drive the signal, must do so at first reset so that both sides of the connection can agree on the value of this signal. Fields:

default_value bool
The default value of the signal we have been assuming.
value bool
The value of the signal.

ArchMsg.Warning.bad_tcu_reset_in

The signal was not driven at reset by the external system, but was driven after we first started using the default value we had assumed. The external system, if it is going to drive the signal, must do so at first reset so that both sides of the connection can agree on the value of this signal. Fields:

default_value bool
The default value of the signal we have been assuming.
value bool
The value of the signal.

ArchMsg.Warning.contig_bit_gives_too_large_region_for_TxSZ

If the contig bit was used then the size of the contig region would be larger than that indicated by TxSZ. Fields:

desc_kind enum
Descriptor kind
el enum
Exception level
input_address unsigned int
Input address
log2_contig_region_size unsigned int
Log2(contig region size in bytes)
output_address unsigned int
Output address
stage_and_level unsigned int
The stage in bits [7:4] and the level in [3:0].
vmid unsigned int
VMID if appropriate

ArchMsg.Warning.contig_bit_has_inconsistent_input_and_output_address

If the contig bit was used then the some bits of the output address held in the descriptor and the input address must match. Fields:

desc_kind enum
Descriptor kind
el enum
Exception level
input_address unsigned int
Input address
log2_contig_region_size unsigned int
Log2(contig region size in bytes)
match_mask unsigned int
Bits that must match.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

output_address unsigned int
: Output address

stage_and_level unsigned int
: The stage in bits [7:4] and the level in [3:0].

vmid unsigned int
: VMID if appropriate

ArchMsg.Warning.fetch_from_memory_type_not_supporting_httu

A descriptor fetch from an HTTU-enabled translation regime to an unsupported memory type was made. Whilst the fetch itself may succeed, if an update to the descriptor was attempted then it would fail. Fields:

address unsigned int
: The address of the descriptor fetch.

desc_inner enum
: Inner cacheability of descriptor

desc_outer enum
: Outer cacheability of descriptor

desc_prot enum
: The instruction/data, bus-ns/s and privileged/user marking

desc_sh enum
: Shareability of descriptor

ssd_ns bool
: Is the Security State Determination of the transaction non-secure.

stage unsigned int
: The stage at which we had a problem.

streamid unsigned int
: The StreamID of the transaction

substreamid unsigned int
: The SubstreamID of the transaction (or ~0u if no substreamid)

trans_id unsigned int
: Transaction id.

ArchMsg.Warning.msi_address_truncated

An MSI was generated, but the address was silently truncated due to the limited downstream address bus width. Fields:

pmcg_index_and_counter unsigned int
: If this is a PMCG interrupt, then the top 16 bits are the PMCG index, the lower 16 bits are counter index.

ras_error_group_id_node_id_and_record_index_in_node unsigned int
: If this is a RAS interrupt then this is the error group record id in bits[23:16], the node_id in [15:8] and the record index in the node in [7:0].

truncated_address unsigned int
: The truncated address.

untruncated_address unsigned int
: The untruncated address.

which enum
: Which MSI this is

ArchMsg.Warning.msi_lost

An MSI was attempted to be sent, but the implementation doesn't support it. Fields:

id unsigned int
: ID of this interrupt transaction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

kind enum
: What kind of interrupt.

ArchMsg.Warning.pmcg_non_secure_world

A commentary on potential mis-programming of the PMCG when transitioning the PMCG from the secure world to the non-secure. Fields:

commentary string
: The commentary.

ArchMsg.Warning.pmcg_programming_violates_security

A commentary on potential mis-programming of the PMCG when transitioning the PMCG between non-secure and secure worlds. This is under the assumption that a non-secure asynchronous agent could be writing to the PMCG. Fields:

commentary string
: The commentary.

ArchMsg.Warning.priq_auto_response_failed_to_find_STE

The PRIQ was going to generate an auto-response, but failed to find an STE and so is returning a Failure message to the EndPoint which should disable the PRI interface of the EndPoint. Fields:

trans_id unsigned int
: The ID of the PPR that is being lost.

why enum
: Why the PPR is being lost.

ArchMsg.Warning.priq_overflow_bad_acking

Indicates that an overflow condition was acknowledged by writing to:- SMMU_PRIQ_CONS.OVACKFLG but an overflow condition did not exist. Fields:

new_ovackflg bool
: The new value of the SMMU_PRIQ_CONS.OVACKFLG.

ArchMsg.Warning.priq_smmuen_forces_effective_priqen_low

If SMMUEN == 0, then the effective value of PRIQEN is 0. This warning is triggered when PRIQEN == 1 && SMMUEN == 0; which may not be what was intended. The PRIQ cannot be active if SMMUEN == 0.

ArchMsg.Warning.sev_lost

A SEV was lost because it isn't supported according to SMMU_IDR0.SEV. Fields:

why enum
: Why the SEV was generated.

ArchMsg.Warning.smmu_pcie_rc_is_in_reset_ignoring_atc_invalidate

A CMD_PRI_RESP or CMD_ATC_INV was sent with a streamid that mapped to a PCIe Root Complex that is in reset. The message will be discarded and act as though completed successfully. Fields:

port_index unsigned int
: The port index (node index) that is in reset but we would have sent it to.

streamid unsigned int
: The StreamID that failed to map.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.smmu_pcie_rc_not_found_for_streamid

A CMD_PRI_RESP or CMD_ATC_INV was sent with a streamid that did not map to a PCIe Root Complex. This might be that the SW used an incorrect StreamID or it might be that the model has not been connected correctly. ATC Invalidate messages complete as though successful and PRI Requests are ignored. Fields:

streamid unsigned int
: The StreamID that failed to map.

ArchMsg.Warning.smmu_pmusnapshot

Something strange happened on the pmusnapshot_req/pmusnapshot_ack inteface. Fields:

pin_index unsigned int
: If the PMCG index corresponds to an array of signals, this is the index in the array, or 0 otherwise.

pmcg_index enum
: The PMCG index.

warning string
: The warning message.

ArchMsg.Warning.warning

These messages are about unusual (but not necessarily incorrect) activity occurring on the SMMU. Fields:

output string
: The stream output

EVENTQ_config

Configuration of interrupt updated. Fields:

address unsigned int
: Address of the MSI.

data unsigned int
: Data payload of the MSI.

irqen bool
: Appropriate bit of SMMU_s_IRQ_CTRL controlling this interrupt.

memattr enum
: Memory type

ns bool
: Non-secure bus attribute.

ra bool
: read-allocate attribute.

sh enum
: Shareability.

smmu_idr0_msi bool
: Are MSIs supported for this security world.

tr bool
: transient attribute

wa bool
: write-allocate attributes.

EVENTQ_irqen

The IRQEN has been changed. Fields:

new_value bool
: The new value of IRQEN.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

EVENTQ_irqen_ack

The IRQEN change has been acked. Fields:

new_value bool
: The new value of IRQEN.

GERROR_config

Configuration of interrupt updated. Fields:

address unsigned int
: Address of the MSI.

data unsigned int
: Data payload of the MSI.

irqen bool
: Appropriate bit of SMMU_s_IRQ_CTRL controlling this interrupt.

memattr enum
: Memory type

ns bool
: Non-secure bus attribute.

ra bool
: read-allocate attribute.

sh enum
: Shareability.

smmu_idr0_msi bool
: Are MSIs supported for this security world.

tr bool
: transient attribute

wa bool
: write-allocate attributes.

GERROR_irqen

The IRQEN has been changed. Fields:

new_value bool
: The new value of IRQEN.

GERROR_irqen_ack

The IRQEN change has been acked. Fields:

new_value bool
: The new value of IRQEN.

PRIQ_config

Configuration of interrupt updated. Fields:

address unsigned int
: Address of the MSI.

data unsigned int
: Data payload of the MSI.

irqen bool
: Appropriate bit of SMMU_s_IRQ_CTRL controlling this interrupt.

memattr enum
: Memory type

ns bool
: Non-secure bus attribute.

ra bool
: read-allocate attribute.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

sh enum
: Shareability.

smmu_idr0_msi bool
: Are MSIs supported for this security world.

tr bool
: transient attribute

wa bool
: write-allocate attributes.

PRIQ_irqen

The IRQEN has been changed. Fields:

new_value bool
: The new value of IRQEN.

PRIQ_irqen_ack

The IRQEN change has been acked. Fields:

new_value bool
: The new value of IRQEN.

SMMU_CR0ACK_SMMUEN_hazarded_by_priq

The SMMU_CR0ACK.SMMUEN cannot acknowledge the change to SMMUEN because there are outstanding PRIQ writes.

SMMU_CR0ACK_SMMUEN_update

The acknowledge to SMMU_CR0.SMMUEN was performed. Fields:

new_value bool
: The new value of the register.

SMMU_CR0_SMMUEN_old_set_complete

A set of transactions associated with the old value of SMMU_CR0.SMMUEN completed. Fields:

last bool
: This is the last set completing.

old_value bool
: The old value of the set completing.

SMMU_CR0_SMMUEN_write

A write to SMMU_CR0.SMMUEN occurred. Fields:

new_value bool
: The new value of the register.

old_value bool
: The old value of the register.

SMMU_GBPA_old_set_complete

A set of transactions associated with the old value of SMMU_GBPA completed. Fields:

last bool
: This is the last set completing.

old_value unsigned int
: The old value of the set completing.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SMMU_GBPA_update

The Update flag to SMMU_GBPA was lowered. Fields:

new_value unsigned int
: The new value of the register.

SMMU_GBPA_write

A write to SMMU_GBPA occurred. Fields:

new_value unsigned int
: The new value of the register.

old_value unsigned int
: The old value of the register.

SMMU_S_CR0ACK_SMMUEN_update

The acknowledge to SMMU_S_CR0.SMMUEN was performed. Fields:

new_value bool
: The new value of the register.

SMMU_S_CR0_SMMUEN_old_set_complete

A set of transactions associated with the old value of SMMU_S_CR0.SMMUEN completed. Fields:

last bool
: This is the last set completing.

old_value bool
: The old value of the set completing.

SMMU_S_CR0_SMMUEN_write

A write to SMMU_S_CR0.SMMUEN occurred. Fields:

new_value bool
: The new value of the register.

old_value bool
: The old value of the register.

SMMU_S_GBPA_old_set_complete

A set of transactions associated with the old value of SMMU_S_GBPA completed. Fields:

last bool
: This is the last set completing.

old_value unsigned int
: The old value of the set completing.

SMMU_S_GBPA_update

The Update flag to SMMU_S_GBPA was lowered. Fields:

new_value unsigned int
: The new value of the register.

SMMU_S_GBPA_write

A write to SMMU_S_GBPA occurred. Fields:

new_value unsigned int
: The new value of the register.

old_value unsigned int
: The old value of the register.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

S_EVENTQ_config

Configuration of interrupt updated. Fields:

address unsigned int
: Address of the MSI.

data unsigned int
: Data payload of the MSI.

irqen bool
: Appropriate bit of SMMU_s_IRQ_CTRL controlling this interrupt.

memattr enum
: Memory type

ns bool
: Non-secure bus attribute.

ra bool
: read-allocate attribute.

sh enum
: Shareability.

smmu_idr0_msi bool
: Are MSIs supported for this security world.

tr bool
: transient attribute

wa bool
: write-allocate attributes.

S_EVENTQ_irqen

The IRQEN has been changed. Fields:

new_value bool
: The new value of IRQEN.

S_EVENTQ_irqen_ack

The IRQEN change has been acked. Fields:

new_value bool
: The new value of IRQEN.

S_GERROR_config

Configuration of interrupt updated. Fields:

address unsigned int
: Address of the MSI.

data unsigned int
: Data payload of the MSI.

irqen bool
: Appropriate bit of SMMU_s_IRQ_CTRL controlling this interrupt.

memattr enum
: Memory type

ns bool
: Non-secure bus attribute.

ra bool
: read-allocate attribute.

sh enum
: Shareability.

smmu_idr0_msi bool
: Are MSIs supported for this security world.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

tr bool
: transient attribute

wa bool
: write-allocate attributes.

S_GERROR_irqen

The IRQEN has been changed. Fields:

new_value bool
: The new value of IRQEN.

S_GERROR_irqen_ack

The IRQEN change has been acked. Fields:

new_value bool
: The new value of IRQEN.

TLB.tlb_commentary

A commentary from the cache about what is going on. Fields:

output string
: The stream output.

TLB.tlb_entry_invalidated

An entry in the cache has been invalidated. Fields:

entry_id unsigned int
: The entry id that is being invalidated.

reason enum
: Why the entry is being invalidated.

atc_inv_nop

The CMD_ATC_INV command is ignored as a NOP. This may be emitted multiple times if the CMD_ATC_INV is being ignored for multiple reasons. Fields:

address unsigned int
: Untranslated Address to invalidate.

cmd_id unsigned int
: Command id, if top-bit is set then was issued from the Non-secure CMDQ.

cons unsigned int
: CONS of the command.

global bool
: Global flag.

log2_size_in_bytes unsigned int
: Log 2 of the size in bytes of the region to invalidate.

streamid unsigned int
: StreamID in the invalidate command.

substreamid unsigned int
: SubstreamID or ~0u if SSValid == false.

why enum
: Why was NOPped

atos_complete_fault

The ATOS operation completed with a fault. Fields:

effective_st1translate bool
: Because of the settings and/or the ATOS type then the effective st1translate can be different.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

fault_faddr unsigned int
: The fault FADDR.

fault_faultcode enum
: The fault code.

fault_reason enum
: The fault reason.

ssd_ns bool
: This is a non-secure ATOS operation.

st1translate bool
: The StreamID/SubstreamID combination should have been translated by stage 1.

st2translate bool
: The translation suffered a stage 2 translation.

state enum
: The final transaction state of the ATOS operation.

vatos_vmid unsigned int
: The VMID of the VATOS operation, or ~0u if not a VATOS operation.

atos_complete_fault_inv_req

The ATOS operation completed, faulted and generated an INV_REQ response. Fields:

ssd_ns bool
: This is a non-secure ATOS operation.

vatos_vmid unsigned int
: The VMID of the VATOS operation, or ~0u if not a VATOS operation.

why enum
: Why the ATOS operation generated an INV_REQ

atos_complete_success

The ATOS operation completed successfully. Fields:

base_addr unsigned int
: The actual base address of region.

effective_st1translate bool
: Because of the settings and/or the ATOS type then the effective st1translate can be different.

par_addr unsigned int
: The PAR.ADDR field.

par_mair unsigned int
: The memory attributes encoded as a MAIR

par_ns bool
: The PAR.NS field, for an SSD-ns request then this will always be 0.

par_sh enum
: Shareability

par_size bool
: The PAR.Size field.

size_in_bytes unsigned int
: The actual size in bytes of the region.

ssd_ns bool
: This is a non-secure ATOS operation.

st1translate bool
: The StreamID/SubstreamID combination should have been translated by stage 1.

st2translate bool
: The translation suffered a stage 2 translation.

state enum
: The final transaction state of the ATOS operation.

vatos_vmid unsigned int
: The VMID of the VATOS operation, or ~0u if not a VATOS operation.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

atos_run_set

The SMMU_s_GATOS_CTRL.RUN field was set to start the ATOS operation. Fields:

ssd_ns bool
: This is a non-secure ATOS operation.

vatos bool
: This is a VATOS operation

atos_starting

The ATOS operation is starting. Fields:

addr unsigned int
: The input address to the ATOS operation.

httui bool
: Inhibit HTTU update.

ind bool
: Instruction Data.

pnu bool
: Privileged not User.

rnw bool
: Read not Write.

ssd_ns bool
: This is a non-secure ATOS operation.

ssec bool
: If this is a secure ATOS operation then this is if it is secure or not.

streamid unsigned int
: The StreamID requested.

substreamid unsigned int
: The SubstreamID requested, or ~0u if no SubstreamID.

type enum
: The requested ATOS type.

vatos_vmid unsigned int
: The VMID of the VATOS operation, or ~0u if not a VATOS operation.

cd_cc.CD_commentary

A commentary from the cache about what is going on. Fields:

output string
: The stream output.

cd_cc.CD_entry_invalidated

An entry in the cache has been invalidated. Fields:

entry_id unsigned int
: The entry id that is being invalidated.

reason enum
: Why the entry is being invalidated.

cd_entry_allocated

An CD entry has been allocated. Fields:

cd string
: A textual description of the CD.

entry_id unsigned int
: The id of the cache entry. Top bit is set if is a debug entry.

ns enum
: For the non-secure world

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

streamid unsigned int
: The StreamID of the entry this will match.

substreamid unsigned int
: The SubstreamID of the entry this will match. This may be zero for transactions without a SubstreamID.

conf_reset_of_SMMU_GBPA_ABORT

The reset value of SMMU_GBPA.ABORT. Fields:

value bool
: The value of the signal.

conf_reset_of_SMMU_S_GBPA_ABORT

The reset value of SMMU_S_GBPA.ABORT. Fields:

value bool
: The value of the signal.

dvm_tlbinvalidate_complete

The DVM TLB Invalidate message completed. Fields:

id unsigned int
: The unique id of this DVM message.

ok bool
: The DVM message was OK.

dvm_tlbinvalidate_received

A DVM message for a TLB Invalidate has been received. Fields:

address unsigned int
: The VA or IPA to use if match_address.

asid unsigned int
: The ASID to match if match_asid.

by_ipa bool
: The operation is for an IPA operation if match_address.

id unsigned int
: The unique id of this DVM message.

ignored enum
: The DVM message was ignored

last_level bool
: The operation is for last level if supported.

match_address bool
: Match the address field.

match_asid bool
: Match the asid field.

match_vmid bool
: Match the vmid field.

num unsigned int
: If a range operation, the NUM field. If a single-address operation this is 0.

prot enum
: The protection level for which this TLB Invalidate will operate on.

security_world enum
: The security world that this will apply to.

smmu_scale unsigned int
: If a range operation, then the SCALE field with the meaning in the SMMU architecture which is different to the PE architecture. If a single-address operation this is 0.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

stage1_only bool
: The operation is for stage 1 only if supported.

tg enum
: If a single-address or address-range operation, then the Translation Granule hint. Address-range operations always supply a Translation Granule.

translation_table_level enum
: The leaf level of the translation table.

vmid unsigned int
: The VMID to match if match_vmid.

found_tlb_entry_has_different_aset

Architecturally, a particular ASID either should be ASET0 or ASET1. However, we have managed to find a TLB entry that has a different ASET than that which we were searching for. This indicates a programming error. You should examine all contexts with this particular ASID/VMID and ensure they are consistent. Fields:

aset unsigned int
: The ASET of this entry.

asid unsigned int
: ASID if appropriate.

index unsigned int
: Index of TLB entry.

input_start_address unsigned int
: The start address of the input range that this matches.

scheme enum
: The tagging scheme used

ssd_ns enum
: Security State

vmid unsigned int
: VMID if appropriate.

gerror_inverted

A GERROR bit was attempted to be inverted to record a fault. Fields:

already_different_to_gerrorn bool
: True if the bit was already different to the corresponding bit in GERRORN and so it was not inverted.

gerrorn unsigned int
: GERRORN register value.

interrupt_potentially_sent bool
: An interrupt is potentially sent, depending on IRQEN.
GERROR(N).MSI_GERROR_ABT_ERR does not send an interupt.

new_gerror unsigned int
: GERROR register value after any invertion.

which enum
: Which bit was attempted to be inverted.

gerrorn_acknowledge

A GERROR was acknowledged by SW. Fields:

which enum
: Which GERROR was acknowledged.

httu_update_abandoned_update

The HTTU update of a descriptor in memory was potentially possible, but it was behind an update that failed to apply cleanly. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

AF enum
: Whether an AF update should/could occur.

DBM enum
: Whether a DBM update should/could occur.

address unsigned int
: Address of HTTU update.

httu_inner enum
: Inner cacheability of descriptor to update

httu_outer enum
: Outer cacheability of descriptor to update

httu_prot enum
: The instruction/data, bus-ns/s and privileged/user marking

httu_sh enum
: Shareability of descriptor to update

mpam_pmg_and_partid unsigned int
: Bits[23:16] are the MPAM PMG. Bits[15:0] are the physical PARTID.

original_descriptor unsigned int
: The original descriptor that the model observed.

stage_and_level unsigned int
: The stage (bits[7:4]) and level (bits[3:0]).

trans_id unsigned int
: The transaction id that this HTTU update is associated with.

what enum
: What this descriptor represents

httu_update_end_update

The attempted update of a descriptor in memory has occurred. Fields:

is_big_endian bool
: The descriptor is big-endian in memory.

original_descriptor unsigned int
: The original descriptor value.

result enum
: The result of the attempt to update.

trans_id unsigned int
: The transaction id that this HTTU update is associated with.

try_to_change_to_descriptor unsigned int
: The new descriptor value that tried to replace the original.

value_that_was_in_memory unsigned int
: The value that the compare-and-swap operation returned as the value that was in memory.

httu_update_not_done

A discretionary HTTU update could occur and the implementation choose not to do it. A mandatory_do_if_linked_performed means that this is a stage 2 leaf descriptor that maps a subsequent stage 1 leaf descriptor whose update is discretionary and if that subsequent discretionary update is going to occur then this update becomes mandatory. Fields:

AF enum
: Whether an AF update should/could occur.

DBM enum
: Whether a DBM update should/could occur.

address unsigned int
: Address of HTTU update.

httu_inner enum
: Inner cacheability of descriptor to update

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

httu_outer enum
: Outer cacheability of descriptor to update

httu_prot enum
: The instruction/data, bus-ns/s and privileged/user marking

httu_sh enum
: Shareability of descriptor to update

mpam_pmg_and_partid unsigned int
: Bits[23:16] are the MPAM PMG. Bits[15:0] are the physical PARTID.

original_descriptor unsigned int
: The original descriptor that the model observed.

stage_and_level unsigned int
: The stage (bits[7:4]) and level (bits[3:0]).

trans_id unsigned int
: The transaction id that this HTTU update is associated with.

what enum
: What this descriptor represents

httu_update_start_update

An HTTU update could occur and the implementation chose to try it. A mandatory_do_if_linked_performed means that this is a stage 2 leaf descriptor that maps a subsequent stage 1 leaf descriptor whose update is discretionary and if that subsequent discretionary update is going to occur then this update becomes mandatory. Fields:

AF enum
: Whether an AF update should/could occur.

DBM enum
: Whether a DBM update should/could occur.

address unsigned int
: Address of HTTU update.

httu_inner enum
: Inner cacheability of descriptor to update

httu_outer enum
: Outer cacheability of descriptor to update

httu_prot enum
: The instruction/data, bus-ns/s and privileged/user marking

httu_sh enum
: Shareability of descriptor to update

is_big_endian bool
: The descriptor will be written to memory as big-endian.

mpam_pmg_and_partid unsigned int
: Bits[23:16] are the MPAM PMG. Bits[15:0] are the physical PARTID.

original_descriptor unsigned int
: The original descriptor that the model observed.

stage_and_level unsigned int
: The stage (bits[7:4]) and level (bits[3:0]).

trans_id unsigned int
: The transaction id that this HTTU update is associated with.

try_to_change_to_descriptor unsigned int
: The new descriptor value that is going to try replace the original.

what enum
: What this descriptor represents

will_do_AF bool
: What the implementation chose to do for the AF update.

will_do_DBM bool
: What the implementation chose to do for the DBM update.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

interrupt_returned

An interrupt/MSI returned from downstream. Fields:

InD_NS_PnU enum
: Instruction/Data, Non-secure/Secure, Privileged/User.

address unsigned int
: Address of the MSI.

data unsigned int
: Data payload of the MSI.

id unsigned int
: ID of this interrupt transaction.

inner_cache enum
: The actual attributes of the access.

kind enum
: What kind of interrupt.

mpam_ns bool
: The security state associated with the MPAM PARTID/PMG.

mpam_pmg_and_partid unsigned int
: Bits[23:16] are the MPAM PMG. Bits[15:0] are the physical PARTID.

msi_valid bool
: MSI will attempt to be sent.

ok bool
: Did the access return OK or an abort?

outer_cache enum
: The actual attributes of the access.

pmcg_index_and_counter unsigned int
: If this is a PMCG interrupt, then the top 16 bits are the PMCG index, the lower 16 bits are counter index.

ras_error_group_id_node_id_and_record_index_in_node unsigned int
: If this is a RAS interrupt then this is the error group record id in bits[23:16], the node_id in [15:8] and the record index in the node in [7:0].

sh enum
: Shareability

interrupt_sent

An interrupt is raised. If it sends an MSI then this is _after_ any device-dependent transform on the architectural attributes and so may differ from what is programmed. Fields:

InD_NS_PnU enum
: Instruction/Data, Non-secure/Secure, Privileged/User.

address unsigned int
: Address of the MSI.

data unsigned int
: Data payload of the MSI.

id unsigned int
: ID of this interrupt transaction.

inner_cache enum
: The actual attributes of the access.

kind enum
: What kind of interrupt.

mpam_ns bool
: The NS state of the MPAM PARTID and MPAM PMG.

mpam_pmg_and_partid unsigned int
: Bits[23:16] are the MPAM PMG. Bits[15:0] are the physical PARTID.

msi_valid bool
: MSI will attempt to be sent.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

outer_cache enum
: The actual attributes of the access.

pmcg_index_and_counter unsigned int
: If this is a PMCG interrupt, then the top 16 bits are the PMCG index, the lower 16 bits are counter index.

ras_error_group_id_node_id_and_record_index_in_node unsigned int
: If this is a RAS interrupt then this is the error group record id in bits[23:16], the node_id in [15:8] and the record index in the node in [7:0].

sh enum
: Shareability

l1cd_cc.L1CD_commentary

A commentary from the cache about what is going on. Fields:

output string
: The stream output.

l1cd_cc.L1CD_entry_invalidated

An entry in the cache has been invalidated. Fields:

entry_id unsigned int
: The entry id that is being invalidated.

reason enum
: Why the entry is being invalidated.

l1cd_entry_allocated

An L1 CD entry has been allocated. Fields:

entry_id unsigned int
: The id of the cache entry. Top bit is set if is a debug entry.

normalised_substreamid unsigned int
: The first SubstreamID of the range of SubstreamIDs that this L1CD entry will match.

ns enum
: For the non-secure world

pa_l2 unsigned int
: The PA of the L2 CD table. This is L2Ptr << 12.

streamid unsigned int
: The StreamID this CD is for.

valid bool
: Is the entry valid.

l1ste_cc.L1STE_commentary

A commentary from the cache about what is going on. Fields:

output string
: The stream output.

l1ste_cc.L1STE_entry_invalidated

An entry in the cache has been invalidated. Fields:

entry_id unsigned int
: The entry id that is being invalidated.

reason enum
: Why the entry is being invalidated.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

l1ste_entry_allocated

An L1 STE entry has been allocated. Fields:

entry_id unsigned int
: The id of the cache entry. Top bit is set if is a debug entry.

first_streamid_of_range unsigned int
: The first StreamID of the range of StreamIDs that this L1STE entry will match.

ns enum
: For the non-secure world

num_entries_in_l2 unsigned int
: The number of entries in the L2 table. This is 2**(Span-1) or 0 if invalid.

pa_l2 unsigned int
: The PA of the L2 ST table. This is L2Ptr << 6 and aligned to the size of the table.

ns_cmd_sync_completed_irq

Non-secure CMD_SYNC completed its IRQ action. Fields:

cmd_id unsigned int
: The ID of the command being issued.

cons_incl_wrap unsigned int
: CONS pointer of CMD_SYNC.

ns_cmd_sync_completed_no_action

Non-secure CMD_SYNC completed, there was no CMD_SYNC action (SIG_NONE). Fields:

cmd_id unsigned int
: The ID of the command being issued.

cons_incl_wrap unsigned int
: CONS pointer of CMD_SYNC.

ns_cmd_sync_completed_sev

Non-secure CMD_SYNC completed its action of SIG_SEV. Fields:

cmd_id unsigned int
: The ID of the command being issued.

cons_incl_wrap unsigned int
: CONS pointer of CMD_SYNC.

ns_cmd_sync_completed_with_error

Non-secure CMD_SYNC completed, however, there was an error associated with the completion of the CMD_SYNC. Fields:

cerror enum
: The error associated with the CMD_SYNC

cmd_id unsigned int
: The ID of the command being issued.

cons_incl_wrap unsigned int
: CONS pointer of CMD_SYNC.

ns_cmd_sync_issued

Non-secure CMD_SYNC has now completed issuing and we now know if something is hazarding the CMD_SYNC from completing. Fields:

cmd_id unsigned int
: The ID of the command being issued.

cons_incl_wrap unsigned int
: CONS pointer of CMD_SYNC.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

hazarded bool
: The CMD_SYNC is currently hazarded from completing.

ns_cmd_sync_issuing

Non-secure CMD_SYNC is starting to issue, we are going to start accounting for which things have to be part of the prior set of things that have to complete before the CMD_SYNC can complete. Fields:

cmd_id unsigned int
: The ID of the command being issued.

cons_incl_wrap unsigned int
: CONS pointer of CMD_SYNC.

ns_cmd_sync_starting_completion_action_irq

Non-secure CMD_SYNC starting IRQ completion action. Fields:

cmd_id unsigned int
: The ID of the command being issued.

cons_incl_wrap unsigned int
: CONS pointer of CMD_SYNC.

msiaddress unsigned int
: The MSI address to use (or 0 if not sending an MSI).

msidata unsigned int
: The MSI data to send.

ns_cmd_sync_starting_completion_action_sev

Non-secure CMD_SYNC starting its completion action of SIG_SEV. Fields:

cmd_id unsigned int
: The ID of the command being issued.

cons_incl_wrap unsigned int
: CONS pointer of CMD_SYNC.

ns_cmdq_cmd_sync_error

The CMD_SYNC can fault if an ATC Invalidate timeout occurs. The timeout will be attached to the next CMD_SYNC on the queue on which the invalidate was launched. Fields:

cerror enum
: The CERROR to attach.

cmd_queue unsigned int
: The top bits indicate the CMDQ id. These bits are attached to all cmd_ids generated for this queue.

ns_cmdq_cmd_sync_error_delivered

The CMD_SYNC can fault if an ATC Invalidate timeout occurs. This is the CMD_SYNC delivering that fault. Fields:

cerror enum
: The CERROR to attach.

cmd_id unsigned int
: Command ID of the CMD_SYNC

ns_cmdq_ecmdq_enack

Trace the changes to ENACK. Fields:

cerror enum
: The error being exposed, if any.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

cmd_queue unsigned int
: The top bits indicate the CMDQ id. These bits are attached to all cmd_ids generated for this queue.

enack bool
: The value of ENACK.

forced_error bool
: The queue was enabled with ERR != ERRACK and so the implementation chooses to trigger the error reporting mechanism.

ns_cmdq_issue

A command is actually being executed. Fields:

cmd_id unsigned int
: Command ID of the command being executed.

cons unsigned int
: CONS of the command.

what string
: What is being executed.

ns_cmdq_pointers_state

The command queue pointers. Fields:

cmd_queue unsigned int
: The top bits indicate the CMDQ id. These bits are attached to all cmd_ids generated for this queue.

disable_fetch bool
: The model has decided that as you programmed PROD/CONS inconsistently that it would not fetch as it has no confidence as to which are commands and which are not.

q_issuable_and_issued_cons unsigned int
: The CONS pointer of the region that is issuable and/or issued (including wrap).

q_issuable_and_issued_prod unsigned int
: The PROD pointer of the region that is issuable and/or issued (including wrap).

q_programmer_cons unsigned int
: The CONS pointer as seen by the programmer (including wrap).

q_programmer_prod unsigned int
: The PROD pointer as seen by the programmer (including wrap).

q_unfetched_cons unsigned int
: The CONS pointer of the region unfetched (including wrap).

q_unfetched_prod unsigned int
: The PROD pointer of the region unfetched (including wrap).

ns_cmdq_state

A description of what the CMDQ can do now. Fields:

can_cerror bool
: There is a CERROR that can be recognised.

can_cmd_sync bool
: There is a CMD_SYNC that can be recognised.

can_fetch bool
: The SMMU can fetch

can_issue bool
: There are fetched commands that are eligible to be issued.

can_update_cons bool
: The programmer view of CMDQ_CONS can be moved to say some have been consumed.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

cmd_queue unsigned int
: The top bits indicate the CMDQ id. These bits are attached to all cmd_ids generated for this queue.

cmd_sync enum
: Why a CMD_SYNC cannot be issued

cmdqen bool
: CMDQEN, when enabled then the CMDQ can fetch commands

cmdqen_1_to_0_unacknowledged bool
: CMDQEN has been set to 0, but we have yet to acknowledge it.

current-real-cerror enum
: The current CERROR that would be visible to the programmer if there was an error.

fetch-disabled_due_to_programmer_error bool
: The PROD/CONS pointer was inconsistent and the model decided to stop processing commands.

fetch-number_of_unfetched_commands unsigned int
: The number of commands that have yet to be fetched from memory, or ~0u if CMDQEN == 0.

fetch-state enum
: The state of the fetch state machine

fetch-too_many_commands_outstanding_so_not_fetching_any_more bool
: If there are too may issued commands, or pending to be issued then we won't fetch any more.

issue-number_of_issuable_commands unsigned int
: The number of commands that have been fetched but not yet issued.

issue-number_of_unissuable_commands unsigned int
: This might include commands behind an illegal command, or behind a CMD_SYNC. Under some circumstances this can include a command representing an external abort.

issue-state enum
: The internal issued state machine state

outstanding_fetches_since_reset unsigned int
: The number of outstanding fetches.

ns_eventq_adding_to_cmd_sync_prior_set

The event created by the specified transaction id must become visible to the programmer before the CMD_SYNC can complete. Fields:

trans_id unsigned int
: The transaction id that caused the event.

ns_eventq_auto_retry_stalled_transaction

A transaction that was stalled but not yet reported on the Event Queue was retried. The retry can happen if the transaction didn't write because it was de-duplicated behind a reported stalling transaction and that stalling transaction was CMD_RESUMEd, or if the queue was unwriteable and the queue is now writeable. Fields:

trans_id unsigned int
: Transaction id of the auto-retried transaction.

why enum
: Why the transaction retried.

ns_eventq_cmd_sync_unhazarded

The CMD_SYNC has been unhazarded as the prior set is empty and as much of it as required is visible in the programmers' view. Fields:

prod_incl_wrap unsigned int
: The prod index including wrap bit that was required to become visible.

prog_prod_incl_wrap unsigned int
: The current programmer-visible prod including wrap bit.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ns_eventq_eventqen_and_eventqenack

Trace the values of EVENTQEN and EVENTQENACK as they change. Fields:

about enum
: Some more information about the queue state.

cons_incl_wrap unsigned int
: The CONS pointer (including wrap bits).

eventqen bool
: EVENTQEN.

eventqenack bool
: EVENTQENACK.

prod_incl_wrap unsigned int
: The PROD pointer (including wrap bits).

ns_eventq_external_abort

A particular eventq record got an external abort. The event will subsequently appear in the ns/s_eventq_lost_event_records trace. Fields:

prod unsigned int
: The prod pointer (excluding wrap bit) of the event record that aborted.

trans_id unsigned int
: The transaction id that produced the event record that aborted.

ns_eventq_lost_event_records

Event records that are lost and never record appear on this trace. Fields:

dword0 unsigned int
: Double word 0 of event.

dword1 unsigned int
: Double word 1 of event.

dword2 unsigned int
: Double word 2 of event.

dword3 unsigned int
: Double word 3 of event.

event enum
: Event code that was thrown away.

prod unsigned int
: The PROD index of the event if appropriate or ~0u otherwise.

reason enum
: The reason the record was lost.

trans_id unsigned int
: The transaction id that this corresponds to.

ns_eventq_overflow_acknowledged

When events are lost then they appear on the eventq_lost_event_records_* trace. This will activate the overflow flag in the corresponding SMMU_s_EVENTQ_PROD register. Software can then acknowledge that flag using the overflow acknowledge flag in SMMU_s_EVENTQ_CONS. Fields:

bad bool
: If true then the programmer has attempted to acknowledge an overflow condition that didn't exist.

count_of_events_lost_due_to_overflow unsigned int
: This is the number of events that were lost because the event queue was full. This does not include any lost because of an EVENTQ_ABT_ERR.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ns_eventq_pending_event_records

Event records that are pending to be dealt with by the eventq state machine. A specific event can be traced multiple times if it becomes part of the prior set of transactions of a CMD_SYNC that has to complete before the CMD_SYNC is allowed to be completed. Fields:

dword0 unsigned int
: Double word 0 of event.

dword1 unsigned int
: Double word 1 of event.

dword2 unsigned int
: Double word 2 of event.

dword3 unsigned int
: Double word 3 of event.

event enum
: Event code.

is_prior_set bool
: The transaction forms part of the prior set of a CMD_SYNC that have to become visible (or thrown away if necessary) before the CMD_SYNC can complete

trans_id unsigned int
: The transaction id that this corresponds to.

ns_eventq_prod_cons_updated

This records when the programmer visible PROD/CONS pointer is updated. More can be visible in memory than are advertised in the programmer-visible PROD pointer. This also traces the latest PROD that could have been advertised. Fields:

mem_prod unsigned int
: The PROD pointer value for all records that are visible in the memory.

num_events_available unsigned int
: The number of events available to the programmer.

prog_cons unsigned int
: The programmer-visible CONS pointer.

prog_prod unsigned int
: The programmer-visible PROD pointer.

what enum
: What changed?

ns_eventq_raising_eventq_abt_err

EVENTQ_ABT_ERR can now be raised.

ns_eventq_removing_from_cmd_sync_prior_set

The event captured by a CMD_SYNC and created by the specified transaction id is now in memory. Fields:

prod unsigned int
: The prod index (no wrap bit) that must become visible to the programmer to see this record. If not head_of_line then the record will not become visible and this field is meaningless.

reason enum
: The reason the event was removed.

trans_id unsigned int
: The transaction id that caused the event.

ns_eventq_stashing_unreported_stalled_transaction_for_auto_retry

A stall event record would have been lost, instead it is stashed away to auto-retry when it can. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

dword0 unsigned int
: Double word 0 of event.

dword1 unsigned int
: Double word 1 of event.

dword2 unsigned int
: Double word 2 of event.

dword3 unsigned int
: Double word 3 of event.

event enum
: Event code that was thrown away.

reason enum
: The reason the record was lost.

trans_id unsigned int
: The transaction id that this corresponds to.

ns_eventq_write_event

An event is committed to being written. Fields:

dword0 unsigned int
: Double word 0 of event.

dword1 unsigned int
: Double word 1 of event.

dword2 unsigned int
: Double word 2 of event.

dword3 unsigned int
: Double word 3 of event.

event enum
: Event code that was thrown away.

prod unsigned int
: The PROD pointer (without wrap) that this event will be written to.

stall_event bool
: Describes an event for a stalled transaction.

trans_id unsigned int
: Transaction ID that generated this event.

ns_eventq_write_event_visible_in_memory

An event is now visible in memory. Fields:

abort bool
: The write aborted.

dword0 unsigned int
: Double word 0 of event.

dword1 unsigned int
: Double word 1 of event.

dword2 unsigned int
: Double word 2 of event.

dword3 unsigned int
: Double word 3 of event.

event enum
: Event code that was thrown away.

prod unsigned int
: The PROD pointer (without wrap) that this event will be written to.

stall_event bool
: Describes an event for a stalled transaction.

trans_id unsigned int
: Transaction ID that generated this event.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

pmcg_irq_config

The interrupt configuration of the Performance Monitor Counter Group (PMCG) changed. Fields:

address unsigned int
: Address of the MSI.

data unsigned int
: Data payload of the MSI.

irqen bool
: SMMU_PMCG_CTRL.IRQEN.

memattr enum
: Memory type

mpam_ns bool
: The NS state of the MPAM PARTID and MPAM PMG

mpam_pmg_and_partid unsigned int
: Bits[23:16] are the MPAM PMG. Bits[15:0] are the physical PARTID.

msi_supported bool
: Are MSIs supported for this security world.

ns bool
: Non-secure bus attribute.

number_of_interrupts_in_flight unsigned int
: The number of interrupts that have been committed to be produced or in flight.

pmcg_index unsigned int
: Index of the PMCG

sh enum
: Shareability.

smmu_pmcg_gmpam_Update bool
: The SMMU_PMCG_GMPAM.Update flag. Only when this is zero are writes predictable.

ssd_ns bool
: True if this PMCG controlled by the non-secure world.

pmcg_irqen

A trace of SMMU_PMCG_IRQ_CTRL.IRQEN. Fields:

new_value bool
: The new value of IRQEN.

pmcg_index unsigned int
: Index of the PMCG.

ssd_ns bool
: True if this PMCG controlled by the non-secure world.

pmcg_irqen_ack

A trace of SMMU_PMCG_IRQ_CTRLACK.IRQEN. Fields:

new_value bool
: The new value of IRQEN.

pmcg_index unsigned int
: Index of the PMCG.

ssd_ns bool
: True if this PMCG controlled by the non-secure world.

pmcg_merging_interrupts

An interrupt was wanted to be generated, but one was already pending so the two were merged together. Fields:

pmcg_index unsigned int
: Index of the PMCG.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ssd_ns bool
: True if this PMCG controlled by the non-secure world.

pmcg_trigger

A PMCG counter has been triggered. Fields:

counter_index unsigned int
: The index of the counter within the PMCG.

event_id unsigned int
: The event id that has been triggered.

ns_event bool
: The security state associated with the event.

pmcg_index unsigned int
: Index of the PMCG.

streamid unsigned int
: The StreamID associated with the event, if there is one.

pmu_active_counter

Traces what active counters are in a PMCG and what streamids it might filter on. Those counters that trace both secure and non-secure StreamIDs or those that are not filtered by StreamID will appear twice, once for each security world. All active counters for a PMCG are traced one after another. Fields:

begin_streamid unsigned int
: The start StreamID to filter on.

counter_index unsigned int
: The counter index within the PMCG.

end_incl_streamid unsigned int
: The end inclusive StreamID to filter on.

evcnt unsigned int
: The current count.

event_id unsigned int
: The event ID to filter.

ns bool
: Are the StreamIDs non-secure?

pmcg_index unsigned int
: The index of the PMCG that this counter belongs to.

pmu_all_counters_in_pmcg_became_inactive

The PMCG was tracing some events and now is not tracing any. Fields:

pmcg_index unsigned int
: The index of the PMCG.

pmu_capture

For some reason, a capture event occurred. Fields:

pmcg_index unsigned int
: The index of the PMCG that the capture occurred on.

why enum
: Why did the capture occur?

pmu_counter_configured_to_use_unsupported_event

An enabled counter was configured to use a unsupported event. Fields:

counter_index unsigned int
: The counter index within the PMCG.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

event_id unsigned int
: The unsupported event id.

pmcg_index unsigned int
: The index of the PMCG that this counter belongs to.

pmu_counter_overflowed

A counter in a particular PMCG overflowed. Fields:

already_overflowed bool
: True if the overflow flag was already set.

capture bool
: True if it captured the other counter values.

counter_index unsigned int
: The counter index within the PMCG.

interrupt bool
: True if going to attempt to generate an interrupt.

interrupt_action enum
: The interrupt action that is going to occur

pmcg_index unsigned int
: The index of the PMCG that this counter belongs to.

ssd_ns bool
: The PMCG is controlled by the Non-secure world.

priq_auto_response

A PRIQ auto response is generated. Fields:

prgindex unsigned int
: The PRG Index of the response.

resp enum
: The response to send.

streamid unsigned int
: The StreamID of the response.

substreamid unsigned int
: The PASID/SubstreamID attached to the response, or ~0u if there is none.

trans_id unsigned int
: The ID of the PPR that is being lost.

why enum
: Why the PPR is being lost.

priq_lost_ppr

The PRIQ is not reporting a PPR into the queue. It may or may not generate an auto-response. Fields:

is_stop_marker bool
: The PPR being lost is a Stop Marker.

resp enum
: What, if any, auto-response will be.

trans_id unsigned int
: The ID of the PPR that is being lost.

why enum
: Why the PPR is being lost

priq_overflow_acking

Indicates that an overflow condition was acknowledged by writing to:-
SMMU_PRIQ_CONS.OVACKFLG. Fields:

new_ovackflg bool
: The new value of the SMMU_PRIQ_CONS.OVACKFLG.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

priq_overflow_asserting

Indicates that we are toggling the SMMU_PRIQ_PROD.OVFLG because we lost a PRI request due to the PRIQ being full and an existing overflow condition does not already exist. Fields:

new_ovflg bool
: The new value of the SMMU_PRIQ_PROD.OVFLG.

trans_id unsigned int
: The transaction ID of the PPR that caused the overflow.

priq_received

A PRIQ Request has been received and is queued waiting processing. This does not mean that it has been written to the PRIQ but has been placed in a pending queue awaiting a decision about what to do about it. Fields:

L bool
: The Last in the Page Request Group (PRG).

R bool
: Read permission is requested.

W bool
: Read permission is requested.

prgindex unsigned int
: The Page Request Group (PRG) index that is being used to identify this request to the EndPoint.

ssd_ns bool
: The SSD of the StreamID. PRIQ should only receive _non-secure_ PRI Requests.

stop_marker bool
: If LRW == 0b100 then this indicates this is a stop marker.

streamid unsigned int
: The StreamID this request is associated with.

substreamid unsigned int
: The SubstreamID (PASID) this request is associated with, of ~0u if no SubstreamID.

substreamid_P bool
: If has a SubstreamID (!= ~0u) then Privileged Mode Requested.

substreamid_X bool
: If has a SubstreamID (!= ~0u) then Execution Requested (R must also be 1).

trans_id unsigned int
: The transaction ID that identifies this request.

untranslated_address unsigned int
: The Untranslated Address that this is requesting.

priq_write_aborted

A PRIQ write aborted. The PRIQ now goes into an error state and will start auto-responding to PRI requests. Fields:

prod_incl_wrap unsigned int
: The PROD including the Wrap bit where we were trying to write.

trans_id unsigned int
: The transaction ID of the PPR that aborted.

priq_write_ok

A PRIQ write completed OK. Fields:

prod_incl_wrap unsigned int
: The PROD including the Wrap bit where we wrote..

trans_id unsigned int
: The transaction ID of the PPR.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

priq_write_start

A PRIQ request has been received and is going to be attempt to be written to the queue. Fields:

L bool
: The Last in the Page Request Group (PRG).

R bool
: Read permission is requested.

W bool
: Read permission is requested.

prgindex unsigned int
: The Page Request Group (PRG) index that is being used to identify this request to the EndPoint.

prod_incl_wrap unsigned int
: PROD position including the wrap bit.

stop_marker bool
: If LRW == 0b100 then this indicates this is a stop marker.

streamid unsigned int
: The StreamID this request is associated with.

substreamid unsigned int
: The SubstreamID (PASID) this request is associated with, of ~0u if no SubstreamID.

substreamid_P bool
: If has a SubstreamID (!= ~0u) then Privileged Mode Requested.

substreamid_X bool
: If has a SubstreamID (!= ~0u) then Execution Requested (R must also be 1).

trans_id unsigned int
: The transaction ID that identifies this request.

untranslated_address unsigned int
: The Untranslated Address that this is requesting.

priqen_state

The trace of various fields that indicate the state of the PRIQ. Fields:

cons_incl_wrap unsigned int
: The value of SMMU_PRIQ.CONS.RD_and_RD_wrap.

number_of_pprs unsigned int
: The number of PPRs as indicated by the CONS/PROD.

number_of_pprs_still_to_deal_with unsigned int
: This is the number of PPRs that are currently waiting to either be written to the PRIQ, or auto-responded to.

number_of_priq_writes_in_flight unsigned int
: The number of writes to the PRIQ that are currently in flight.

ovackflg bool
: The OVACKFLG which if different to OVFLG is used to indicate that the PRIQ overflowed.

ovflg bool
: The OVFLG which if different to OVACKFLG is used to indicate that the PRIQ overflowed.

priq_abt_err bool
: There is an active SMMU_GERROR{N}.PRIQ_ABT_ERR.

priqen bool
: The value of SMMU_CR0.PRIQEN. The _effective_ value is 0 if SMMUEN == 0.

priqenack bool
: The value of SMMU_CR0ACK.PRIQEN.

prod_incl_wrap unsigned int
: The value of SMMU_PRIQ.CONS.WR_and_WR_wrap.

queue_disabled_due_to_prior_programming_error bool
: The queue was disabled as the programmer got CONS/PROD into an inconsistent state. The model will disable the PRIQ until SW disables and re-enables the queue via SMMU_CR0.PRIQEN.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

smmuen bool
: The value of SMMU_CR0.SMMUEN. If this is 0 then the effective PRIQEN is 0.

smmuenack bool
: The value of SMMU_CR0.SMMUEN. If this is 0 then the effective PRIQEN is 0.

table_size_in_elements unsigned int
: The size of the table in the number of items it can hold.

ptw_read

Page Table Walk (read). This is the result of the physical access that the SMMU is making. Fields:

abort enum
: Non-zero if the access aborted/failed.

adomain enum
: The actual attributes of the access that was used, after IMP DEF mangling.

aprot enum
: The actual attributes of the access that was used, after IMP DEF mangling.

data unsigned int
: The data fetch if it didn't abort.

desckind enum
: The kind of descriptor

inner_cache enum
: The actual attributes of the access that was used, after IMP DEF mangling.

input_address unsigned int
: Input address this lookup is for (after TBI normalisation).

ns bool
: Non-Secure

outer_cache enum
: The actual attributes of the access that was used, after IMP DEF mangling.

pa_address unsigned int
: Physical address.

ssd_ns enum
: Security state of stream

streamid unsigned int
: The StreamID this is for.

substreamid unsigned int
: The SubstreamID this is for.

trans_id unsigned int
: Transaction id

ttb_grain_stage_and_level unsigned int
: Which TTB used in bits[31:24]. Which Grain size used in bit[23:16]:- 0 -- 4 KiB 1 -- 16 KiB 2 -- 64 KiB 3 -- 64 KiB ARMv8.2 where descriptor bits [15:12] for bits [51:48] of output address Stage in bits[15:8]. Level in bits[7:0].

ptw_read_st1_invalid_long_descriptor

Page Table Walk (read). The descriptor that the PTW fetched is invalid. Fields:

desckind enum
: The kind of descriptor

input_address unsigned int
: Input address this lookup is for (after TBI normalisation).

ns bool
: Non-Secure

pa_address unsigned int
: Physical address.

ssd_ns enum
: Security state of stream

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

streamid unsigned int
: The StreamID this is for.

substreamid unsigned int
: The SubstreamID this is for.

trans_id unsigned int
: Transaction id

ttb_grain_stage_and_level unsigned int
: Which TTB used in bits[31:24]. Which Grain size used in bit[23:16]:- 0 -- 4 KiB 1 -- 16 KiB 2 -- 64 KiB 3 -- 64 KiB ARMv8.2 where descriptor bits [15:12] for bits [51:48] of output address Stage in bits[15:8]. Level in bits[7:0].

ptw_read_st1_leaf_long_descriptor

Page Table Walk (read). The descriptor that the PTW fetched is a block or page and this is the decode. Fields:

AF bool
: Access Flag.

AP21 enum
: The access permissions.

AttrIndx210 unsigned int
: The attribute index into the MAIR0/1.

DBM bool
: Dirty Bit Modifier. This only has meaning if running in AArch64 mode.

NS bool
: The encoding is for non-secure if this is a secure fetch.

PXN bool
: Privileged eXecute Never.

SH10 enum
: The shareability.

XN bool
: eXecute Never.

contiguous bool
: Contiguous hint.

desckind enum
: The kind of descriptor

hwu_pbha unsigned int
: Top four bits are appropriate CD.HWU*, bottom bits[62:59] of descriptor. Page Based Hardware Attributes: only valid on a bit-per-bit basis enabled by CD.HWU0*/CD.HWU1*.

input_address unsigned int
: Input address this lookup is for (after TBI normalisation).

nG bool
: not Global.

nT bool
: SMMUv3.2: If true then do not cache this entry in such a way that it can cause a TLB conflict abort (F_TLB_CONFLICT). The entry must produce a consistent result.

ns bool
: Non-Secure

output_address unsigned int
: Output address.

pa_address unsigned int
: Physical address.

ssd_ns enum
: Security state of stream

streamid unsigned int
: The StreamID this is for.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

substreamid unsigned int
: The SubstreamID this is for.

trans_id unsigned int
: Transaction id

ttb_grain_stage_and_level unsigned int
: Which TTB used in bits[31:24]. Which Grain size used in bit[23:16]:- 0 -- 4 KiB 1 -- 16 KiB 2 -- 64 KiB 3 -- 64 KiB ARMv8.2 where descriptor bits [15:12] for bits [51:48] of output address Stage in bits[15:8]. Level in bits[7:0].

ptw_read_st1_table_long_descriptor

Page Table Walk (read). The descriptor that the PTW fetched is a Table, this decodes the fields. Fields:

APTable enum
: Remove permissions independently of subsequent descriptors.

NSTable bool
: The next level table descriptor is forced to non-secure.

PXNTable bool
: Force PXN independently of subsequent descriptors.

TableAddress unsigned int
: Address of the next table.

XNTable bool
: Force XN independently of subsequent descriptors.

desckind enum
: The kind of descriptor

input_address unsigned int
: Input address this lookup is for (after TBI normalisation).

ns bool
: Non-Secure

pa_address unsigned int
: Physical address.

ssd_ns enum
: Security state of stream

streamid unsigned int
: The StreamID this is for.

substreamid unsigned int
: The SubstreamID this is for.

trans_id unsigned int
: Transaction id

ttb_grain_stage_and_level unsigned int
: Which TTB used in bits[31:24]. Which Grain size used in bit[23:16]:- 0 -- 4 KiB 1 -- 16 KiB 2 -- 64 KiB 3 -- 64 KiB ARMv8.2 where descriptor bits [15:12] for bits [51:48] of output address Stage in bits[15:8]. Level in bits[7:0].

ptw_read_st2_invalid_descriptor

Page Table Walk (read). The descriptor that the PTW fetched is invalid. Fields:

desckind enum
: The kind of descriptor

input_address unsigned int
: Input address this lookup is for (after TBI normalisation).

ns bool
: Non-Secure

pa_address unsigned int
: Physical address.

ssd_ns enum
: Security state of stream

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

streamid unsigned int
: The StreamID this is for.

substreamid unsigned int
: The SubstreamID this is for.

trans_id unsigned int
: Transaction id

ttb_grain_stage_and_level unsigned int
: Which TTB used in bits[31:24]. Which Grain size used in bit[23:16]:- 0 -- 4 KiB 1 -- 16 KiB 2 -- 64 KiB 3 -- 64 KiB ARMv8.2 where descriptor bits [15:12] for bits [51:48] of output address Stage in bits[15:8]. Level in bits[7:0].

ptw_read_st2_leaf_descriptor

Page Table Walk (read). The descriptor that the PTW fetched is a block or page and this is the decode. Fields:

AF bool
: Access Flag.

DBM bool
: Dirty Bit Modifier. This only has meaning if running in AArch64 mode.

HAP21 enum
: The access permissions.

MemAttr3_0 enum
: The memory attributes.

SH10 enum
: The shareability.

XN enum
: eXecute Never.

contiguous bool
: Contiguous hint.

desckind enum
: The kind of descriptor

input_address unsigned int
: Input address this lookup is for (after TBI normalisation).

nT bool
: SMMUv3.2: If true then do not cache this entry in such a way that it can cause a TLB conflict abort (F_TLB_CONFLICT). The entry must produce a consistent result.

ns bool
: Non-Secure

output_address unsigned int
: Output address.

pa_address unsigned int
: Physical address.

s2hwu_pbha unsigned int
: Top four bits are STE.S2HWU*, bottom bits[62:59] of descriptor. Page Based Hardware Attributes: only valid on a bit-per-bit basis enabled by STE.S2HWU*.

ssd_ns enum
: Security state of stream

streamid unsigned int
: The StreamID this is for.

substreamid unsigned int
: The SubstreamID this is for.

trans_id unsigned int
: Transaction id

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ttb_grain_stage_and_level unsigned int
: Which TTB used in bits[31:24]. Which Grain size used in bit[23:16]:- 0 -- 4 KiB 1 -- 16 KiB 2 -- 64 KiB 3 -- 64 KiB ARMv8.2 where descriptor bits [15:12] for bits [51:48] of output address Stage in bits[15:8]. Level in bits[7:0].

ptw_read_st2_table_descriptor

Page Table Walk (read). The descriptor that the PTW fetched is a Table, this decodes the fields. Fields:

APTable enum
: Remove permissions independently of subsequent descriptors.

PXNTable bool
: Force PXN independently of subsequent descriptors.

TableAddress unsigned int
: Address of the next table.

XNTable bool
: Force XN independently of subsequent descriptors.

desckind enum
: The kind of descriptor

input_address unsigned int
: Input address this lookup is for (after TBI normalisation).

ns bool
: Non-Secure

pa_address unsigned int
: Physical address.

ssd_ns enum
: Security state of stream

streamid unsigned int
: The StreamID this is for.

substreamid unsigned int
: The SubstreamID this is for.

trans_id unsigned int
: Transaction id

ttb_grain_stage_and_level unsigned int
: Which TTB used in bits[31:24]. Which Grain size used in bit[23:16]:- 0 -- 4 KiB 1 -- 16 KiB 2 -- 64 KiB 3 -- 64 KiB ARMv8.2 where descriptor bits [15:12] for bits [51:48] of output address Stage in bits[15:8]. Level in bits[7:0].

raw_register_end_read

The raw register read transaction. This is the transaction as directed to the register port. Fields:

id unsigned int
: Id of this transaction.

ok bool
: Was the access OK? The bus response will always be OK, but is the register access reasonable.

read_data unsigned int
: The data read.

raw_register_end_write

The raw register write transaction. This is the transaction as directed to the register port. Fields:

id unsigned int
: Id of this transaction.

ok bool
: Was the access OK? The bus response will always be OK, but is the register access reasonable.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

raw_register_start_read

The raw register read transaction. This is the transaction as directed to the register port. Fields:

address unsigned int
: Address of transaction.

id unsigned int
: Id of this transaction. Top bit is set if is a debug transaction.

ns enum
: Is the transaction non-secure?

number_of_beats unsigned int
: Number of beats.

size_of_beat_in_bytes unsigned int
: Size of each beat in bytes.

raw_register_start_write

The raw register write transaction. This is the transaction as directed to the register port. Fields:

address unsigned int
: Address of transaction

id unsigned int
: Id of this transaction. Top bit is set if is a debug transaction.

ns enum
: Is the transaction non-secure?

number_of_beats unsigned int
: Number of beats.

size_of_beat_in_bytes unsigned int
: Size of each beat in bytes.

write_data unsigned int
: The data to write.

register_disallowed_read_string

A text representation of the read of a register that was disallowed. Fields:

out string
: The text description of the register value read.

register_disallowed_write_string

A text representation of the write of a register write that was disallowed. Fields:

in string
: The text description of the register value written.

register_read_reserved

A text representation of an access to a register address that is reserved. Fields:

in string
: The text description of the register value.

register_read_string

A text representation of the read of a register. Fields:

out string
: The text description of the register value read.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

register_write_reserved

A text representation of an access to a register address that is reserved or a write.to a RES0 field in a register. Fields:

in `string`
: The text description of the register value.

register_write_string

A text representation of the write of a register. Fields:

in `string`
: The text description of the register value written.

s_cmd_sync_completed_irq

Secure CMD_SYNC completed its IRQ action. Fields:

cmd_id `unsigned int`
: The ID of the command being issued.

cons_incl_wrap `unsigned int`
: CONS pointer of CMD_SYNC.

s_cmd_sync_completed_no_action

Secure CMD_SYNC completed, there was no CMD_SYNC action (SIG_NONE). Fields:

cmd_id `unsigned int`
: The ID of the command being issued.

cons_incl_wrap `unsigned int`
: CONS pointer of CMD_SYNC.

s_cmd_sync_completed_sev

Secure CMD_SYNC completed its action of SIG_SEV. Fields:

cmd_id `unsigned int`
: The ID of the command being issued.

cons_incl_wrap `unsigned int`
: CONS pointer of CMD_SYNC.

s_cmd_sync_completed_with_error

Secure CMD_SYNC completed, however, there was an error associated with the completion of the CMD_SYNC. Fields:

cerror `enum`
: The error associated with the CMD_SYNC

cmd_id `unsigned int`
: The ID of the command being issued.

cons_incl_wrap `unsigned int`
: CONS pointer of CMD_SYNC.

s_cmd_sync_issued

Secure CMD_SYNC has now completed issuing and we now know if something is hazarding the CMD_SYNC from completing. Fields:

cmd_id `unsigned int`
: The ID of the command being issued.

cons_incl_wrap `unsigned int`
: CONS pointer of CMD_SYNC.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

hazarded bool
: The CMD_SYNC is currently hazarded from completing.

s_cmd_sync_issuing

Secure CMD_SYNC is starting to issue, we are going to start accounting for which things have to be part of the prior set of things that have to complete before the CMD_SYNC can complete. Fields:

cmd_id unsigned int
: The ID of the command being issued.

cons_incl_wrap unsigned int
: CONS pointer of CMD_SYNC.

s_cmd_sync_starting_completion_action_irq

Secure CMD_SYNC starting IRQ completion action. Fields:

cmd_id unsigned int
: The ID of the command being issued.

cons_incl_wrap unsigned int
: CONS pointer of CMD_SYNC.

msiaddress unsigned int
: The MSI address to use (or 0 if not sending an MSI).

msidata unsigned int
: The MSI data to send.

s_cmd_sync_starting_completion_action_sev

Secure CMD_SYNC starting its completion action of SIG_SEV. Fields:

cmd_id unsigned int
: The ID of the command being issued.

cons_incl_wrap unsigned int
: CONS pointer of CMD_SYNC.

s_cmdq_cmd_sync_error

The CMD_SYNC can fault if an ATC Invalidate timeout occurs. The timeout will be attached to the next CMD_SYNC on the queue on which the invalidate was launched. Fields:

cerror enum
: The CERROR to attach.

cmd_queue unsigned int
: The top bits indicate the CMDQ id. These bits are attached to all cmd_ids generated for this queue.

s_cmdq_cmd_sync_error_delivered

The CMD_SYNC can fault if an ATC Invalidate timeout occurs. This is the CMD_SYNC delivering that fault. Fields:

cerror enum
: The CERROR to attach.

cmd_id unsigned int
: Command ID of the CMD_SYNC

s_cmdq_ecmdq_enack

Trace the changes to ENACK. Fields:

cerror enum
: The error being exposed, if any.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

cmd_queue unsigned int
: The top bits indicate the CMDQ id. These bits are attached to all cmd_ids generated for this queue.

enack bool
: The value of ENACK.

forced_error bool
: The queue was enabled with ERR != ERRACK and so the implementation chooses to trigger the error reporting mechanism.

s_cmdq_issue

A command is actually being executed. Fields:

cmd_id unsigned int
: Command ID of the command being executed.

cons unsigned int
: CONS of the command.

what string
: What is being executed.

s_cmdq_pointers_state

The command queue pointers. Fields:

cmd_queue unsigned int
: The top bits indicate the CMDQ id. These bits are attached to all cmd_ids generated for this queue.

disable_fetch bool
: The model has decided that as you programmed PROD/CONS inconsistently that it would not fetch as it has no confidence as to which are commands and which are not.

q_issuable_and_issued_cons unsigned int
: The CONS pointer of the region that is issuable and/or issued (including wrap).

q_issuable_and_issued_prod unsigned int
: The PROD pointer of the region that is issuable and/or issued (including wrap).

q_programmer_cons unsigned int
: The CONS pointer as seen by the programmer (including wrap).

q_programmer_prod unsigned int
: The PROD pointer as seen by the programmer (including wrap).

q_unfetched_cons unsigned int
: The CONS pointer of the region unfetched (including wrap).

q_unfetched_prod unsigned int
: The PROD pointer of the region unfetched (including wrap).

s_cmdq_state

A description of what the CMDQ can do now. Fields:

can_cerror bool
: There is a CERROR that can be recognised.

can_cmd_sync bool
: There is a CMD_SYNC that can be recognised.

can_fetch bool
: The SMMU can fetch

can_issue bool
: There are fetched commands that are eligible to be issued.

can_update_cons bool
: The programmer view of CMDQ_CONS can be moved to say some have been consumed.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

cmd_queue unsigned int
: The top bits indicate the CMDQ id. These bits are attached to all cmd_ids generated for this queue.

cmd_sync enum
: Why a CMD_SYNC cannot be issued

cmdqen bool
: CMDQEN, when enabled then the CMDQ can fetch commands

cmdqen_1_to_0_unacknowledged bool
: CMDQEN has been set to 0, but we have yet to acknowledge it.

current-real-cerror enum
: The current CERROR that would be visible to the programmer if there was an error.

fetch-disabled_due_to_programmer_error bool
: The PROD/CONS pointer was inconsistent and the model decided to stop processing commands.

fetch-number_of_unfetched_commands unsigned int
: The number of commands that have yet to be fetched from memory, or ~0u if CMDQEN == 0.

fetch-state enum
: The state of the fetch state machine

fetch-too_many_commands_outstanding_so_not_fetching_any_more bool
: If there are too may issued commands, or pending to be issued then we won't fetch any more.

issue-number_of_issuable_commands unsigned int
: The number of commands that have been fetched but not yet issued.

issue-number_of_unissuable_commands unsigned int
: This might include commands behind an illegal command, or behind a CMD_SYNC. Under some circumstances this can include a command representing an external abort.

issue-state enum
: The internal issued state machine state

outstanding_fetches_since_reset unsigned int
: The number of outstanding fetches.

s_eventq_adding_to_cmd_sync_prior_set

The event created by the specified transaction id must become visible to the programmer before the CMD_SYNC can complete. Fields:

trans_id unsigned int
: The transaction id that caused the event.

s_eventq_auto_retry_stalled_transaction

A transaction that was stalled but not yet reported on the Event Queue was retried. The retry can happen if the transaction didn't write because it was de-duplicated behind a reported stalling transaction and that stalling transaction was CMD_RESUMEd, or if the queue was unwriteable and the queue is now writeable. Fields:

trans_id unsigned int
: Transaction id of the auto-retried transaction.

why enum
: Why the transaction retried.

s_eventq_cmd_sync_unhazarded

The CMD_SYNC has been unhazarded as the prior set is empty and as much of it as required is visible in the programmers' view. Fields:

prod_incl_wrap unsigned int
: The prod index including wrap bit that was required to become visible.

prog_prod_incl_wrap unsigned int
: The current programmer-visible prod including wrap bit.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

s_eventq_eventqen_and_eventqenack

Trace the values of EVENTQEN and EVENTQENACK as they change. Fields:

about enum
: Some more information about the queue state.

cons_incl_wrap unsigned int
: The CONS pointer (including wrap bits).

eventqen bool
: EVENTQEN.

eventqenack bool
: EVENTQENACK.

prod_incl_wrap unsigned int
: The PROD pointer (including wrap bits).

s_eventq_external_abort

A particular eventq record got an external abort. The event will subsequently appear in the ns/ s_eventq_lost_event_records trace. Fields:

prod unsigned int
: The prod pointer (excluding wrap bit) of the event record that aborted.

trans_id unsigned int
: The transaction id that produced the event record that aborted.

s_eventq_lost_event_records

Event records that are lost and never record appear on this trace. Fields:

dword0 unsigned int
: Double word 0 of event.

dword1 unsigned int
: Double word 1 of event.

dword2 unsigned int
: Double word 2 of event.

dword3 unsigned int
: Double word 3 of event.

event enum
: Event code that was thrown away.

prod unsigned int
: The PROD index of the event if appropriate or ~0u otherwise.

reason enum
: The reason the record was lost.

trans_id unsigned int
: The transaction id that this corresponds to.

s_eventq_overflow_acknowledged

When events are lost then they appear on the eventq_lost_event_records_* trace. This will activate the overflow flag in the corresponding SMMU_s_EVENTQ_PROD register. Software can then acknowledge that flag using the overflow acknowledge flag in SMMU_s_EVENTQ_CONS. Fields:

bad bool
: If true then the programmer has attempted to acknowledge an overflow condition that didn't exist.

count_of_events_lost_due_to_overflow unsigned int
: This is the number of events that were lost because the event queue was full. This does not include any lost because of an EVENTQ_ABT_ERR.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

s_eventq_pending_event_records

Event records that are pending to be dealt with by the eventq state machine. A specific event can be traced multiple times if it becomes part of the prior set of transactions of a CMD_SYNC that has to complete before the CMD_SYNC is allowed to be completed. Fields:

dword0 unsigned int
: Double word 0 of event.

dword1 unsigned int
: Double word 1 of event.

dword2 unsigned int
: Double word 2 of event.

dword3 unsigned int
: Double word 3 of event.

event enum
: Event code.

is_prior_set bool
: The transaction forms part of the prior set of a CMD_SYNC that have to become visible (or thrown away if necessary) before the CMD_SYNC can complete

trans_id unsigned int
: The transaction id that this corresponds to.

s_eventq_prod_cons_updated

This records when the programmer visible PROD/CONS pointer is updated. More can be visible in memory than are advertised in the programmer-visible PROD pointer. This also traces the latest PROD that could have been advertised. Fields:

mem_prod unsigned int
: The PROD pointer value for all records that are visible in the memory.

num_events_available unsigned int
: The number of events available to the programmer.

prog_cons unsigned int
: The programmer-visible CONS pointer.

prog_prod unsigned int
: The programmer-visible PROD pointer.

what enum
: What changed?

s_eventq_raising_eventq_abt_err

EVENTQ_ABT_ERR can now be raised.

s_eventq_removing_from_cmd_sync_prior_set

The event captured by a CMD_SYNC and created by the specified transaction id is now in memory. Fields:

prod unsigned int
: The prod index (no wrap bit) that must become visible to the programmer to see this record. If not head_of_line then the record will not become visible and this field is meaningless.

reason enum
: The reason the event was removed.

trans_id unsigned int
: The transaction id that caused the event.

s_eventq_stashing_unreported_stalled_transaction_for_auto_retry

A stall event record would have been lost, instead it is stashed away to auto-retry when it can. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

dword0 unsigned int
: Double word 0 of event.

dword1 unsigned int
: Double word 1 of event.

dword2 unsigned int
: Double word 2 of event.

dword3 unsigned int
: Double word 3 of event.

event enum
: Event code that was thrown away.

reason enum
: The reason the record was lost.

trans_id unsigned int
: The transaction id that this corresponds to.

s_eventq_write_event

An event is committed to being written. Fields:

dword0 unsigned int
: Double word 0 of event.

dword1 unsigned int
: Double word 1 of event.

dword2 unsigned int
: Double word 2 of event.

dword3 unsigned int
: Double word 3 of event.

event enum
: Event code that was thrown away.

prod unsigned int
: The PROD pointer (without wrap) that this event will be written to.

stall_event bool
: Describes an event for a stalled transaction.

trans_id unsigned int
: Transaction ID that generated this event.

s_eventq_write_event_visible_in_memory

An event is now visible in memory. Fields:

abort bool
: The write aborted.

dword0 unsigned int
: Double word 0 of event.

dword1 unsigned int
: Double word 1 of event.

dword2 unsigned int
: Double word 2 of event.

dword3 unsigned int
: Double word 3 of event.

event enum
: Event code that was thrown away.

prod unsigned int
: The PROD pointer (without wrap) that this event will be written to.

stall_event bool
: Describes an event for a stalled transaction.

trans_id unsigned int
: Transaction ID that generated this event.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

s_gerror_inverted

A GERROR bit was attempted to be inverted to record a fault. Fields:

already_different_to_gerrorn bool
: True if the bit was already different to the corresponding bit in GERRORN and so it was not inverted.

gerrorn unsigned int
: GERRORN register value.

interrupt_potentially_sent bool
: An interrupt is potentially sent, depending on IRQEN.
GERROR(N).MSI_GERROR_ABT_ERR does not send an interupt.

new_gerror unsigned int
: GERROR register value after any invertion.

which enum
: Which bit was attempted to be inverted.

s_gerrorn_acknowledge

A GERROR was acknowledged by SW. Fields:

which enum
: Which GERROR was acknowledged.

sec_override

Controls non-secure accesses to some registers. Fields:

value bool
: The value of the signal.

sev

Send a SEV. Fields:

why enum
: Why the SEV was generated.

smmu_atc_inv

The CMD_ATC_INV command is sent. Fields:

address unsigned int
: Untranslated Address to invalidate.

cmd_id unsigned int
: Command id, if top-bit is set then was issued from the Non-secure CMDQ.

cons unsigned int
: CONS of the command.

global bool
: Global flag.

log2_size_in_bytes unsigned int
: Log 2 of the size in bytes of the region to invalidate.

ns_number_in_flight unsigned int
: Number of ATC invalidates in flight issued from the Non-secure CMDQ.

ns_number_waiting_to_go_out unsigned int
: The total number of ATC invalidates waiting to be pushed out from the Non-secure CMDQ.

s_number_in_flight unsigned int
: Number of ATC invalidates in flight issued from the Secure CMDQ.

s_number_waiting_to_go_out unsigned int
: The total number of ATC invalidates waiting to be pushed out from the Secure CMDQ.

streamid unsigned int
: StreamID in the invalidate command.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

substreamid unsigned int
: SubstreamID or ~0u if SSValid == false.

smmu_atc_inv_completed

The CMD_ATC_INV command completed. Fields:

cmd_id unsigned int
: Command id, if top-bit is set then was issued from the Non-secure CMDQ.

ns_number_in_flight unsigned int
: Number of ATC invalidates in flight now this one has completed, that were issued by the Non-secure CMDQ.

ns_number_waiting_to_go_out unsigned int
: The total number of ATC invalidates waiting to be pushed out from the Non-secure CMDQ.

response enum
: The response.

s_number_in_flight unsigned int
: Number of ATC invalidates in flight now this one has completed, that were issued by the Secure CMDQ.

s_number_waiting_to_go_out unsigned int
: The total number of ATC invalidates waiting to be pushed out from the Secure CMDQ.

smmu_atc_inv_end

An ATC invalidate request has been started. Fields:

PASID_global bool
: If has a SubstreamID (PASID) then if it is 'global' or not.

address unsigned int
: The Untranslated Address to use.

log2_size_in_bytes unsigned int
: Log2 of the size of the region in bytes to invalidate.

response enum
: The response to the ATC invalidate

streamid unsigned int
: The StreamID to use.

substreamid unsigned int
: The SubstreamID to use (or ~0u if not used). This corresponds to the PASID.

smmu_atc_inv_start

An ATC invalidate request has been started. Fields:

PASID_global bool
: If has a SubstreamID (PASID) then if it is 'global' or not.

address unsigned int
: The Untranslated Address to use.

log2_size_in_bytes unsigned int
: Log2 of the size of the region in bytes to invalidate.

streamid unsigned int
: The StreamID to use.

substreamid unsigned int
: The SubstreamID to use (or ~0u if not used). This corresponds to the PASID.

smmu_ats_initial

This is the initial ATS request. Fields:

ia unsigned int
: Input address

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

max_number_of_replies unsigned int
: The maximum number of replies allowed to return

no_write bool
: The NW (no write flag) of the ATS request. If clear then the requester is going to do a write.

pasid_execute_requested bool
: If there is a PASID (substreamid != ~0u) then this represents the requester asking for execution permissions.

pasid_privileged_mode_requested bool
: If there is a PASID (substreamid != ~0u) then this represents the requester asking for the privileged mode's permissions.

ssd_ns bool
: Incoming SSD, only non-secure ATS requests are legal.

streamid unsigned int
: StreamID of the ATS request.

substreamid unsigned int
: SubstreamID (which is identical to the PASID) of the ATS request. If no PASID-prefix is sent then this is ~0u

tbu unsigned int
: Translation Buffer Unit number

smmu_ats_reply_failure

This is an ATS reply indicating failure. Fields:

event enum
: Equivalent event number that would have been generated for an equivalent ordinary transaction.

failure enum
: What is the failure response code?

state enum
: The transaction state of the successfully ATS request.

smmu_ats_reply_success

This is an ATS reply, typically the SMMU will only return a single response, even if the requester indicated it could accept more replies. NOTE that the SMMU responds with 'success' in some cases when a fault is encountered and RW==0. Fields:

N bool
: Non-snooped access. If one then the requester must clear the NoSnoop bit on transactions, unless otherwise enabled in a Function-specific manner.

P bool
: Privileged mode. These permissions related to privileged mode.

RWX enum
: Read/Write/Execute

U bool
: Untranslated access. If one, and RW !=0 then use UntranslatedAccesses for the allowed accesses by RW(X).

inner enum
: The inner cacheability attributes to use for TranslatedAccesses

input_address unsigned int
: Input address of the ATS request

instcfg enum
: The STE.INSTCFG field

outer enum
: The outer cacheability attributes to use for TranslatedAccesses

privcfg enum
: The STE.PRIVCFG field

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

shareability unsigned int
: The shareability to use for TranslatedAccesses

size unsigned int
: The size of the region covered by this translation.

state enum
: The transaction state of the successfully ATS request.

translated_address unsigned int
: If RW!=0 && U != 0, then the Translated Address that a TranslatedAccess can be made with

smmu_final_transaction

This is the transaction group request to remap has completed one set of remapping. For ordinary transactions, this represents a bundle of transactions with the same attributes but different addresses within a certain range around the address. If it stalls then it will report through this trace source, stall (stag_if_stalling != ~0u) and when resume will issue another smmu_initial_transaction as it undergoes remapping again. This trace source can also represent part of the process for ATOS/ATS or finding STE.PPAR for PRI requests that need to be auto-responded to. Fields:

axmmuflow enum
: The AxMMUFLOW for this transaction group. stallable -- allow the transaction to be stalled if configured. translated-access -- is a PCIe Translated Access non-stallable -- do not stall transaction even if configured. transfault-flow -- do not report for Translation Related Faults, but tell client device to use a page request mechanism instead (for example PRI).

cmo_point enum
: The point associated with the CMO, if applicable

input_address unsigned int
: The input address of the transaction group.

input_inner enum
: Inner cacheability for the input attributes.

input_outer enum
: Outer cacheability for the input attributes.

input_prot enum
: The instruction/data, bus-ns/s and privileged/user marking

input_sh enum
: Shareability for the input attributes.

ipa_address unsigned int
: The IPA of the transaction.

kind enum
: The kind of access this transaction group that this represents.

mpam_pmg_and_partid unsigned int
: Bits[23:16] are the MPAM PMG. Bits[15:0] are the physical PARTID.

operation enum
: The kind of operation that this represents.

output_address unsigned int
: The input address of the transaction group.

output_inner enum
: Inner cacheability for the output attributes.

output_outer enum
: Outer cacheability for the output attributes.

output_prot enum
: The instruction/data, bus-ns/s and privileged/user marking

output_sh enum
: Shareability for the output attributes.

output_vmid unsigned int
: The output VMID/GBPA.IMPDEF or ~0u if not valid.

size_of_region_in_bytes unsigned int
: An imp def size of region for which this translation is valid for.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ssd_ns enum
: The security state of the transaction

stag_if_stalling unsigned int
: This is the STAG used by the transaction if it is going to stall. It is ~0u if it is not going to stall.

state enum
: The final transaction state.

streamid unsigned int
: The StreamID of the transction.

substreamid unsigned int
: The SubstreamID of the transaction or 0xFFFFffff if no SubstreamID.

tbu unsigned int
: Translation Buffer Unit number.

trans_id unsigned int
: Transaction id. Top bit set if debug transaction.

smmu_initial_transaction

This is the transaction group request to remap is going to start one set of remapping. For ordinary transactions, this represents a bundle of transactions with the same attributes but different addresses within a certain range around the address. A stalling transaction will report through this trace source when it unstalls. This trace source can also represent part of the process for ATOS/ATS or finding STE.PPAR for PRI requests that need to be auto-responded to. Fields:

axmmuflow enum
: The AxMMUFLOW for this transaction group. stallable -- allow the transaction to be stalled if configured. translated-access -- is a PCIe Translated Access non-stallable -- do not stall transaction even if configured. transfault-flow -- do not report for Translation Related Faults, but tell client device to use a page request mechanism instead (for example PRI).

cmo_point enum
: The point associated with the CMO, if applicable

input_address unsigned int
: The input address of the transaction group.

input_inner enum
: Inner cacheability for the input attributes.

input_outer enum
: Outer cacheability for the input attributes.

input_prot enum
: The instruction/data, bus-ns/s and privileged/user marking

input_sh enum
: Shareability for the input attributes.

kind enum
: The kind of access this transaction group that this represents.

operation enum
: The kind of operation that this represents.

ssd_ns enum
: The security state of the transaction

streamid unsigned int
: The StreamID of the transction.

substreamid unsigned int
: The SubstreamID of the transaction or 0xFFFFffff if no SubstreamID.

tbu unsigned int
: Translation Buffer Unit number.

trans_id unsigned int
: Transaction id. Top bit set if debug transaction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

smmu_pmusnapshot_ack

Acknowledge the pmusnapshot_req to indicate the snapshot has occurred. This is part of a four-phase handshake. Fields:

pin_index unsigned int
: The pin index of the signal, or ~0u if not relevant.

pmcg_index enum
: The PMCG index.

value bool
: The value of the signal.

smmu_pmusnapshot_req

Take a snap shot of the PMU values as though SMMU_PMCG_CAPR.CAPTURE had been written. This is part of a four-phase handshake. Fields:

pin_index unsigned int
: The pin index of the signal, or ~0u if not relevant.

pmcg_index enum
: The PMCG index.

value bool
: The value of the signal.

smmu_poison_tw_data

Poison data has been returned to a table walk transaction. Fields:

bitmap_of_poison unsigned int
: The bitmap of which beats of the transaction where poisoned.

is_cas bool
: True if this is a compare-and-swap operation.

ns bool
: Non-secure state of the bus transaction.

number_of_64bit_beats unsigned int
: Number of beats this transaction fetched.

paddress unsigned int
: Physical address of the table walk transaction.

ras_group_id unsigned int
: If non-~0u then is the RAS group that we wish to attach the error to. Otherwise will use the IMP DEF policy.

ras_record_index_in_group unsigned int
: If non-~0u then is the RAS record index in the group that we wish to attach the error to. Otherwise will use the IMP DEF policy.

what enum
: What table walk was being performed.

smmu_pri_resp

The CMD_PRI_RESP command is queued to be sent to the PCIe system. Fields:

cmd_id unsigned int
: Command id

cons unsigned int
: CONS of the command.

prgindex unsigned int
: PRG Index.

response enum
: If has a SubstreamID (PASID) then if it is 'global' or not.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

streamid unsigned int
: StreamID in the invalidate command.

substreamid unsigned int
: SubstreamID or ~0u if SSValid == false.

smmu_pri_resp_nop

The CMD_PRI_RESP command was NOPped. Fields:

cmd_id unsigned int
: Command id

cons unsigned int
: CONS of the command.

prgindex unsigned int
: PRG Index.

response enum
: If has a SubstreamID (PASID) then if it is 'global' or not.

streamid unsigned int
: StreamID in the invalidate command.

substreamid unsigned int
: SubstreamID or ~0u if SSValid == false.

why enum
: Why the CMD_PRI_RESP was NOPped.

smmu_priq_resp_fake_return

A PRIQ Response is posted to the PCIe subsystem and so has no acknowledgement that it is received. However, in the model then we artificially know when the the PRIQ Response has been delivered to the PCIe subsystem, even if the ATC has not yet acted on it. Fields:

prgindex unsigned int
: The PRG Index this request corresponds to.

streamid unsigned int
: The StreamID to use.

substreamid unsigned int
: The SubstreamID to use (or ~0u if not used). This corresponds to the PASID.

what enum
: What happened.

smmu_priq_resp_start

A PRIQ Response has been posted to the PCIe subsystem. As the response is posted then there is no way of knowing when it is received by the EndPoint. Fields:

prgindex unsigned int
: The PRG Index this request corresponds to.

response enum
: If has a SubstreamID (PASID) then if it is 'global' or not.

streamid unsigned int
: The StreamID to use.

substreamid unsigned int
: The SubstreamID to use (or ~0u if not used). This corresponds to the PASID.

smmu_thread_wait_wake

Traces a thread's wait/wake status. Fields:

current_ticks unsigned int
: The current tick count of simulated time.

event enum
: What is happening to this thread.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

thread_index unsigned int
: The ID of this thread.

ticks unsigned int
: If the event relates to a time then this is held in this field. Otherwise, 0.

stall_transaction

A transaction is about to stall. Fields:

stag unsigned int
: STAG.

streamid unsigned int
: StreamID.

streamid_ns bool
: The StreamID was for the non-secure world.

trans_id unsigned int
: Transaction id.

stall_transaction_inhibited_by_STALL_MAX

A transaction is about to stall but the maximum number of transactions have stalled and we can't report this one to the event queue (even if non-full). Fields:

streamid unsigned int
: StreamID.

streamid_ns bool
: The StreamID was for the non-secure world.

trans_id unsigned int
: Transaction id.

stall_transaction_resuming

A stalled transaction is resuming. Fields:

stag unsigned int
: STAG if appropriate, or if was inhibited by STALL_MAX then 0xFAFA.

stallresult enum
: What the transaction resumed to do.

streamid unsigned int
: StreamID.

streamid_ns bool
: The StreamID was for the non-secure world.

trans_id unsigned int
: Transaction id.

start_ptw_read

Page Table Walk (read). This is the start of the physical access that the SMMU is making. Fields:

adomain enum
: The actual attributes of the access that was used, after IMP DEF mangling.

aprot enum
: The actual attributes of the access that was used, after IMP DEF mangling.

desckind enum
: The kind of descriptor

inner_cache enum
: The actual attributes of the access that was used, after IMP DEF mangling.

input_address unsigned int
: Input address this lookup is for (after TBI normalisation).

mpam_pmg_and_partid unsigned int
: Bits[23:16] are the MPAM PMG. Bits[15:0] are the physical PARTID.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ns bool
: Non-Secure

outer_cache enum
: The actual attributes of the access that was used, after IMP DEF mangling.

pa_address unsigned int
: Physical address.

ssd_ns enum
: Security state of stream

streamid unsigned int
: The StreamID this is for.

substreamid unsigned int
: The SubstreamID this is for.

trans_id unsigned int
: Transaction id

ttb_grain_stage_and_level unsigned int
: Which TTB used in bits[31:24]. Which Grain size used in bit[23:16]:- 0 -- 4 KiB 1 -- 16 KiB 2 -- 64 KiB 3 -- 64 KiB ARMv8.2 where descriptor bits [15:12] for bits [51:48] of output address Stage in bits[15:8]. Level in bits[7:0].

ste_cc.STE_commentary

A commentary from the cache about what is going on. Fields:

output string
: The stream output.

ste_cc.STE_entry_invalidated

An entry in the cache has been invalidated. Fields:

entry_id unsigned int
: The entry id that is being invalidated.

reason enum
: Why the entry is being invalidated.

ste_entry_allocated

An STE entry has been allocated. Fields:

entry_id unsigned int
: The id of the cache entry. Top bit is set if is a debug entry.

ns enum
: For the non-secure world

ste string
: A textual description of the STE.

streamid unsigned int
: The StreamID of the entry this will match.

sup_btm

The system supports BTM. Fields:

value bool
: The value of the signal.

sup_cohacc

The system supports cohacc. Fields:

value bool
: The value of the signal.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

sup_httu

The system supports HTTU. Fields:

value bool
: The value of the signal.

sup_sev

The system supports SEV. Fields:

value bool
: The value of the signal.

tbu0_reset_in

The reset signal of the TBU0. Fields:

value bool
: The value of the signal.

tcu_reset_in

The reset signal. Fields:

value bool
: The value of the signal.

tlb_entry_allocated

A TLB entry has been allocated. Fields:

aset unsigned int
: The ASET of this entry.

asid unsigned int
: ASID if appropriate.

index unsigned int
: Index of TLB entry.

input_end_incl_address unsigned int
: The end inclusive address of the input range that this matches.

input_start_address unsigned int
: The start address of the input range that this matches.

output_end_incl_address unsigned int
: The end inclusive address of the output range.

output_start_address unsigned int
: The start address of the output range.

scheme enum
: The tagging scheme used

ssd_ns enum
: Security State

tbi bool
: Was the entry formed using Top Byte Ignore (TBI).

vmid unsigned int
: VMID if appropriate.

tlb_info_tlb_entries_overlap

A TLB entry was inserted into the TLB and it overlaps an existing entry. This isn't a problem as it was inserted in such a way that it architecturally works. Fields:

end_address_of_new_entry unsigned int
: End address of new entry.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

end_address_of_old_entry unsigned int
End address of old entry.
how_inserted enum
How the entry was inserted.
index_of_new_entry unsigned int
Index of new entry.
index_of_old_entry unsigned int
Index of old entry.
start_address_of_new_entry unsigned int
Start address of new entry.
start_address_of_old_entry unsigned int
Start address of old entry.

verbose_commentary

This is a verbose commentary on the translation process the SMMU is performing. Fields:

output string
The stream output.

warning_MSI_EVENTQ_address_out_of_range_of_oas

The MSI address is configured outside of the range of the downstream address bus size. Fields:

address unsigned int
The untruncated address of the MSI.

warning_MSI_GERROR_address_out_of_range_of_oas

The MSI address is configured outside of the range of the downstream address bus size. Fields:

address unsigned int
The untruncated address of the MSI.

warning_MSI_PRIQ_address_out_of_range_of_oas

The MSI address is configured outside of the range of the downstream address bus size. Fields:

address unsigned int
The untruncated address of the MSI.

warning_MSI_S_EVENTQ_address_out_of_range_of_oas

The MSI address is configured outside of the range of the downstream address bus size. Fields:

address unsigned int
The untruncated address of the MSI.

warning_MSI_S_GERROR_address_out_of_range_of_oas

The MSI address is configured outside of the range of the downstream address bus size. Fields:

address unsigned int
The untruncated address of the MSI.

warning_MSI_pmcg_address_out_of_range_of_oas

The MSI Address of the Performance Monitor Counter Group (PMCG) is out of range of the OAS and so will be silently truncated. Fields:

address unsigned int
The untruncated address of the MSI.
pmcg_index unsigned int
Index of the PMCG.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ssd_ns bool
: True if this PMCG controlled by the non-secure world.

warning_discarding_interrupt_EVENTQ_as_irqen_low

Interrupt generation is turned off by SMMU_IRQ_CTRL.EVENTQ_IRQEN.

warning_discarding_interrupt_GERROR_as_irqen_low

Interrupt generation is turned off by SMMU_IRQ_CTRL.GERROR_IRQEN.

warning_discarding_interrupt_PRIQ_as_irqen_low

Interrupt generation is turned off by SMMU_S_IRQ_CTRL.PRIQ_IRQEN.

warning_discarding_interrupt_S_EVENTQ_as_irqen_low

Interrupt generation is turned off by SMMU_S_IRQ_CTRL.EVENTQ_IRQEN.

warning_discarding_interrupt_S_GERROR_as_irqen_low

Interrupt generation is turned off by SMMU_S_IRQ_CTRL.GERROR_IRQEN.

warning_gerrorn_bad_acknowledge

A GERROR was acknowledged by SW in the GERRORN register that did not have an active error. The result is UNPREDICTABLE if this will generate an interrupt or not. Fields:

which enum
: Which GERROR was acknowledge when there was no active error.

warning_reg_after_doesnt_match_written_value

A write occurred that tried to set bits in a register, that for one reason or another, failed to get written. Fields:

desc string
: The textual description of what happened.

warning_s_gerrorn_bad_acknowledge

A GERROR was acknowledged by SW in the GERRORN register that did not have an active error. The result is UNPREDICTABLE if this will generate an interrupt or not. Fields:

which enum
: Which GERROR was acknowledge when there was no active error.

what_going_to_do_with_terminated_event

A terminating transaction has produced an event, this tells you what the model is going to do with the event. Fields:

CD.S bool
: The CD.S field if available.

S2 bool
: The event is related to Stage 2.

STE.S1STALLD bool
: The STE.S1STALLD field if available.

STE.S2S bool
: The STE.S2S field if available.

aborts bool
: The transaction will abort.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

axmmuflow enum
: The AxMMUFLOW for this transaction group. stallable -- allow the transaction to be stalled if configured. translated-access -- is a PCIe Translated Access non-stallable -- do not stall transaction even if configured. transfault-flow -- do not report for Translation Related Faults, but tell client device to use a page request mechanism instead (for example PRI).

is_tr_fault bool
: Is a Translation Related fault.

reports bool
: The transaction will attempt to report.

ssd_ns bool
: The transaction is classified as SSD non-secure.

supports_stall_model bool
: The implementation supports the stall model.

trans_id unsigned int
: The transaction id.

why_abort_decision enum
: The reason why the transaction aborted/did not abort.

why_report_decision enum
: The reason why the transaction reported/did not report.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.86 MSIRewriter - trace

This section describes the trace sources.

ArchMsg.Warning.msi_abort

A rewritten MSI was aborted by downstream. Fields:

data unsigned int
: The data value of the MSI.

device_id unsigned int
: The DeviceID presented to the rewriter.

ns bool
: The transaction is non-secure.

original_address unsigned int
: The original address of the transaction.

rewritten_address unsigned int
: The rewritten address of the transaction.

rewritten_data unsigned int
: The rewritten data.

ArchMsg.Warning.msi_strange

A rewritten MSI was translated, but it is not to 'device' memory and so is somewhat suspicious. Fields:

data unsigned int
: The data value of the MSI.

device_id unsigned int
: The DeviceID presented to the rewriter.

ns bool
: The transaction is non-secure.

original_address unsigned int
: The original address of the transaction.

rewritten_address unsigned int
: The rewritten address of the transaction.

rewritten_data unsigned int
: The rewritten data.

rewritten

A transaction that was attempted to be rewritten from the GITS_TRANSLATER register to GITS_TRANSLATE64R. Fields:

action enum
: What we are doing with this transaction.

data unsigned int
: The data value of the MSI.

device_id unsigned int
: The DeviceID presented to the rewriter.

ns bool
: The transaction is non-secure.

original_address unsigned int
: The original address of the transaction.

rewritten_address unsigned int
: The rewritten address of the transaction.

rewritten_data unsigned int
: The rewritten data.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.87 Mali_G51 - trace

This section describes the trace sources.

INFO_AccessInReset

GPU Register %{READ:(WRITE|READ)} to address: %{ADDR} FAILED as GPU is being reset. Fields:

ADDR `unsigned int`
: Address of the accessed register.

READ `bool`
: Transaction type is read or write.

INFO_IrqGpuControl

IRQ signal received %{IRQ_ID}(%{IRQ_NAME}) state %{IRQ_STATE:(SET|CLEAR)}. Fields:

IRQ_ID `unsigned int`
: The IRQ id.

IRQ_NAME `string`
: The IRQ name

IRQ_STATE `bool`
: The IRQ state

INFO_IrqJobControl

IRQ signal received %{IRQ_ID}(%{IRQ_NAME}) state %{IRQ_STATE:(SET|CLEAR)}. Fields:

IRQ_ID `unsigned int`
: The IRQ id.

IRQ_NAME `string`
: The IRQ name

IRQ_STATE `bool`
: The IRQ state

INFO_IrqMmuControl

IRQ signal received %{IRQ_ID}(%{IRQ_NAME}) state %{IRQ_STATE:(SET|CLEAR)}. Fields:

IRQ_ID `unsigned int`
: The IRQ id.

IRQ_NAME `string`
: The IRQ name

IRQ_STATE `bool`
: The IRQ state

INFO_ReadRegister

GPU Register Offset: %{REG_OFFSET} Value: %{VALUE} RegName: %{REG_NAME}. Fields:

REG_NAME `string`
: The name of the register read.

REG_OFFSET `unsigned int`
: The address of the register read.

VALUE `unsigned int`
: Read Value from the register.

INFO_Reset

GPU Reset %{STATE:(START|END)}. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

STATE bool
: Reset state is start or end.

INFO_WriteRegister

GPU Register Offset: %{REG_OFFSET} Updated: From value %{VALUE} To value %{UPDATED_VALUE} RegName: %{REG_NAME}. Fields:

REG_NAME string
: The name of the register updated.

REG_OFFSET unsigned int
: The address of the register updated.

UPDATED_VALUE unsigned int
: New Value in the register.

VALUE unsigned int
: Old Value in the register.

WARN_AccessToUnimplementedRegister

GPU Register %{READ:(WRITE|READ)} to address: %{ADDR} FAILED as the register is unimplemented. Fields:

ADDR unsigned int
: Address of the accessed register.

READ bool
: Transaction type is read or write.

WARN_ReadToWriteOnlyRegister

GPU Register Read to address: %{ADDR} FAILED as the register is write-only RegName: %{REG_NAME}. Fields:

ADDR unsigned int
: Address of the accessed register.

REG_NAME string
: The name of the register updated.

WARN_WriteToReadOnlyRegister

GPU Register Write to address: %{ADDR} FAILED as the register is read-only RegName: %{REG_NAME}. Fields:

ADDR unsigned int
: Address of the accessed register.

REG_NAME string
: The name of the register updated.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.88 Mali_G71 - trace

This section describes the trace sources.

INFO_AccessInReset

GPU Register %{READ:(WRITE|READ)} to address: %{ADDR} FAILED as GPU is being reset. Fields:

ADDR `unsigned int`
: Address of the accessed register.

READ `bool`
: Transaction type is read or write.

INFO_IrqGpuControl

IRQ signal received %{IRQ_ID}(%{IRQ_NAME}) state %{IRQ_STATE:(SET|CLEAR)}. Fields:

IRQ_ID `unsigned int`
: The IRQ id.

IRQ_NAME `string`
: The IRQ name

IRQ_STATE `bool`
: The IRQ state

INFO_IrqJobControl

IRQ signal received %{IRQ_ID}(%{IRQ_NAME}) state %{IRQ_STATE:(SET|CLEAR)}. Fields:

IRQ_ID `unsigned int`
: The IRQ id.

IRQ_NAME `string`
: The IRQ name

IRQ_STATE `bool`
: The IRQ state

INFO_IrqMmuControl

IRQ signal received %{IRQ_ID}(%{IRQ_NAME}) state %{IRQ_STATE:(SET|CLEAR)}. Fields:

IRQ_ID `unsigned int`
: The IRQ id.

IRQ_NAME `string`
: The IRQ name

IRQ_STATE `bool`
: The IRQ state

INFO_ReadRegister

GPU Register Offset: %{REG_OFFSET} Value: %{VALUE} RegName: %{REG_NAME}. Fields:

REG_NAME `string`
: The name of the register read.

REG_OFFSET `unsigned int`
: The address of the register read.

VALUE `unsigned int`
: Read Value from the register.

INFO_Reset

GPU Reset %{STATE:(START|END)}. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

STATE bool
: Reset state is start or end.

INFO_WriteRegister

GPU Register Offset: %{REG_OFFSET} Updated: From value %{VALUE} To value %{UPDATED_VALUE} RegName: %{REG_NAME}. Fields:

REG_NAME string
: The name of the register updated.

REG_OFFSET unsigned int
: The address of the register updated.

UPDATED_VALUE unsigned int
: New Value in the register.

VALUE unsigned int
: Old Value in the register.

WARN_AccessToUnimplementedRegister

GPU Register %{READ:(WRITE|READ)} to address: %{ADDR} FAILED as the register is unimplemented. Fields:

ADDR unsigned int
: Address of the accessed register.

READ bool
: Transaction type is read or write.

WARN_ReadToWriteOnlyRegister

GPU Register Read to address: %{ADDR} FAILED as the register is write-only RegName: %{REG_NAME}. Fields:

ADDR unsigned int
: Address of the accessed register.

REG_NAME string
: The name of the register updated.

WARN_WriteToReadOnlyRegister

GPU Register Write to address: %{ADDR} FAILED as the register is read-only RegName: %{REG_NAME}. Fields:

ADDR unsigned int
: Address of the accessed register.

REG_NAME string
: The name of the register updated.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.89 Mali_G72 - trace

This section describes the trace sources.

INFO_AccessInReset

GPU Register %{READ:(WRITE|READ)} to address: %{ADDR} FAILED as GPU is being reset. Fields:

ADDR `unsigned int`
: Address of the accessed register.

READ `bool`
: Transaction type is read or write.

INFO_IrqGpuControl

IRQ signal received %{IRQ_ID}(%{IRQ_NAME}) state %{IRQ_STATE:(SET|CLEAR)}. Fields:

IRQ_ID `unsigned int`
: The IRQ id.

IRQ_NAME `string`
: The IRQ name

IRQ_STATE `bool`
: The IRQ state

INFO_IrqJobControl

IRQ signal received %{IRQ_ID}(%{IRQ_NAME}) state %{IRQ_STATE:(SET|CLEAR)}. Fields:

IRQ_ID `unsigned int`
: The IRQ id.

IRQ_NAME `string`
: The IRQ name

IRQ_STATE `bool`
: The IRQ state

INFO_IrqMmuControl

IRQ signal received %{IRQ_ID}(%{IRQ_NAME}) state %{IRQ_STATE:(SET|CLEAR)}. Fields:

IRQ_ID `unsigned int`
: The IRQ id.

IRQ_NAME `string`
: The IRQ name

IRQ_STATE `bool`
: The IRQ state

INFO_ReadRegister

GPU Register Offset: %{REG_OFFSET} Value: %{VALUE} RegName: %{REG_NAME}. Fields:

REG_NAME `string`
: The name of the register read.

REG_OFFSET `unsigned int`
: The address of the register read.

VALUE `unsigned int`
: Read Value from the register.

INFO_Reset

GPU Reset %{STATE:(START|END)}. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

STATE bool
: Reset state is start or end.

INFO_WriteRegister

GPU Register Offset: %{REG_OFFSET} Updated: From value %{VALUE} To value %{UPDATED_VALUE} RegName: %{REG_NAME}. Fields:

REG_NAME string
: The name of the register updated.

REG_OFFSET unsigned int
: The address of the register updated.

UPDATED_VALUE unsigned int
: New Value in the register.

VALUE unsigned int
: Old Value in the register.

WARN_AccessToUnimplementedRegister

GPU Register %{READ:(WRITE|READ)} to address: %{ADDR} FAILED as the register is unimplemented. Fields:

ADDR unsigned int
: Address of the accessed register.

READ bool
: Transaction type is read or write.

WARN_ReadToWriteOnlyRegister

GPU Register Read to address: %{ADDR} FAILED as the register is write-only RegName: %{REG_NAME}. Fields:

ADDR unsigned int
: Address of the accessed register.

REG_NAME string
: The name of the register updated.

WARN_WriteToReadOnlyRegister

GPU Register Write to address: %{ADDR} FAILED as the register is read-only RegName: %{REG_NAME}. Fields:

ADDR unsigned int
: Address of the accessed register.

REG_NAME string
: The name of the register updated.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.90 Mali_G76 - trace

This section describes the trace sources.

INFO_AccessInReset

GPU Register %{READ:(WRITE|READ)} to address: %{ADDR} FAILED as GPU is being reset. Fields:

ADDR `unsigned int`
: Address of the accessed register.

READ `bool`
: Transaction type is read or write.

INFO_IrqGpuControl

IRQ signal received %{IRQ_ID}(%{IRQ_NAME}) state %{IRQ_STATE:(SET|CLEAR)}. Fields:

IRQ_ID `unsigned int`
: The IRQ id.

IRQ_NAME `string`
: The IRQ name

IRQ_STATE `bool`
: The IRQ state

INFO_IrqJobControl

IRQ signal received %{IRQ_ID}(%{IRQ_NAME}) state %{IRQ_STATE:(SET|CLEAR)}. Fields:

IRQ_ID `unsigned int`
: The IRQ id.

IRQ_NAME `string`
: The IRQ name

IRQ_STATE `bool`
: The IRQ state

INFO_IrqMmuControl

IRQ signal received %{IRQ_ID}(%{IRQ_NAME}) state %{IRQ_STATE:(SET|CLEAR)}. Fields:

IRQ_ID `unsigned int`
: The IRQ id.

IRQ_NAME `string`
: The IRQ name

IRQ_STATE `bool`
: The IRQ state

INFO_ReadRegister

GPU Register Offset: %{REG_OFFSET} Value: %{VALUE} RegName: %{REG_NAME}. Fields:

REG_NAME `string`
: The name of the register read.

REG_OFFSET `unsigned int`
: The address of the register read.

VALUE `unsigned int`
: Read Value from the register.

INFO_Reset

GPU Reset %{STATE:(START|END)}. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

STATE bool
: Reset state is start or end.

INFO_WriteRegister

GPU Register Offset: %{REG_OFFSET} Updated: From value %{VALUE} To value %{UPDATED_VALUE} RegName: %{REG_NAME}. Fields:

REG_NAME string
: The name of the register updated.

REG_OFFSET unsigned int
: The address of the register updated.

UPDATED_VALUE unsigned int
: New Value in the register.

VALUE unsigned int
: Old Value in the register.

WARN_AccessToUnimplementedRegister

GPU Register %{READ:(WRITE|READ)} to address: %{ADDR} FAILED as the register is unimplemented. Fields:

ADDR unsigned int
: Address of the accessed register.

READ bool
: Transaction type is read or write.

WARN_ReadToWriteOnlyRegister

GPU Register Read to address: %{ADDR} FAILED as the register is write-only RegName: %{REG_NAME}. Fields:

ADDR unsigned int
: Address of the accessed register.

REG_NAME string
: The name of the register updated.

WARN_WriteToReadOnlyRegister

GPU Register Write to address: %{ADDR} FAILED as the register is read-only RegName: %{REG_NAME}. Fields:

ADDR unsigned int
: Address of the accessed register.

REG_NAME string
: The name of the register updated.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.91 MemoryMappedCounterModule - trace

This section describes the trace sources.

CHECKPOINT_MESSAGE

Report error messages from the checkpointing process. Fields:

message string
: Message contents.

CHECKPOINT_RESTORE_END

Checkpoint restore completed.

CHECKPOINT_RESTORE_START

Checkpoint restore about to start.

CHECKPOINT_SAVE_END

Checkpoint save completed.

CHECKPOINT_SAVE_START

Checkpoint save about to start.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.92 MessageHandlingUnitV2 - trace

This section describes the trace sources.

RECV_ArchMsg.Warning.MemoryMapped_AccessAborted

Fields:

OFFSET unsigned int
: Offset of address within the MHU.

RECV_ArchMsg.Warning.MemoryMapped_ReadReserved

Fields:

OFFSET unsigned int
: Offset of address within the MHU.

RECV_ArchMsg.Warning.MemoryMapped_ReadWriteOnlyReg

Fields:

OFFSET unsigned int
: Offset of address within the MHU.

REG_NAME string
: Name of the register being read.

RECV_ArchMsg.Warning.MemoryMapped_WriteReadOnlyReg

Fields:

OFFSET unsigned int
: Offset of address within the MHU.

REG_NAME string
: Name of the register being written.

VALUE unsigned int
: Value written.

RECV_ArchMsg.Warning.MemoryMapped_WriteReserved

Fields:

OFFSET unsigned int
: Offset of address within the MHU.

VALUE unsigned int
: Value written.

RECV_MemoryMapped_Read

Trace read from a memory-mapped register. Fields:

OFFSET unsigned int
: Offset of address within the MHU.

REG_NAME string
: Name of the register being read.

VALUE unsigned int
: Value returned.

RECV_MemoryMapped_Write

Trace write to a memory-mapped register. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

OFFSET unsigned int
Offset of address within the MHU.
REG_NAME string
Name of the register being written.
UPDATED_VALUE unsigned int
New value read back from the register; for write-only registers this will be equal to VALUE.
VALUE unsigned int
Value written.

SEND_ArchMsg.Warning.MemoryMapped_AccessAborted

Fields:

OFFSET unsigned int
Offset of address within the MHU.

SEND_ArchMsg.Warning.MemoryMapped_ReadReserved

Fields:

OFFSET unsigned int
Offset of address within the MHU.

SEND_ArchMsg.Warning.MemoryMapped_ReadWriteOnlyReg

Fields:

OFFSET unsigned int
Offset of address within the MHU.
REG_NAME string
Name of the register being read.

SEND_ArchMsg.Warning.MemoryMapped_WriteReadOnlyReg

Fields:

OFFSET unsigned int
Offset of address within the MHU.
REG_NAME string
Name of the register being written.
VALUE unsigned int
Value written.

SEND_ArchMsg.Warning.MemoryMapped_WriteReserved

Fields:

OFFSET unsigned int
Offset of address within the MHU.
VALUE unsigned int
Value written.

SEND_MemoryMapped_Read

Trace read from a memory-mapped register. Fields:

OFFSET unsigned int
Offset of address within the MHU.
REG_NAME string
Name of the register being read.
VALUE unsigned int
Value returned.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

SEND_MemoryMapped_Write

Trace write to a memory-mapped register. Fields:

OFFSET `unsigned int`
: Offset of address within the MHU.

REG_NAME `string`
: Name of the register being written.

UPDATED_VALUE `unsigned int`
: New value read back from the register; for write-only registers this will be equal to VALUE.

VALUE `unsigned int`
: Value written.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.93 NonVolatileCounter - trace

This section describes the trace sources.

CHECKPOINT_MESSAGE

Report error messages from the checkpointing process. Fields:

message string
: Message contents.

CHECKPOINT_RESTORE_END

Checkpoint restore completed.

CHECKPOINT_RESTORE_START

Checkpoint restore about to start.

CHECKPOINT_SAVE_END

Checkpoint save completed.

CHECKPOINT_SAVE_START

Checkpoint save about to start.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.94 PL011_Uart - trace

This section describes the trace sources.

CHECKPOINT_MESSAGE

Report error messages from the checkpointing process. Fields:

message string
: Message contents.

CHECKPOINT_RESTORE_END

Checkpoint restore completed.

CHECKPOINT_RESTORE_START

Checkpoint restore about to start.

CHECKPOINT_SAVE_END

Checkpoint save completed.

CHECKPOINT_SAVE_START

Checkpoint save about to start.

pl011_character_in

A character was received. Fields:

char unsigned int
: The character received.

fifo_overflow bool
: If true, then the character was lost as the fifo was full.

tick unsigned int
: The count of ticks from simulation start that the UART has received.

pl011_character_out

A character was written to the UART to output. Fields:

char unsigned int
: The output character.

fifo_overflow bool
: If true, then the character was lost as the fifo was full.

tick unsigned int
: The count of ticks from simulation start that the UART has received.

pl011_line_buffered_out

The buffered output of character_out. The buffer is flushed when any control character is received. The control characters do not form part of the buffer. The buffer's size is limited to 255 characters and truncation is indicated by an appended '...' . Fields:

buffer string
: The line buffer.

tick unsigned int
: The count of ticks from simulation start that the UART has received at the point at which it receives the control character that flushes the buffer.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.95 PL11x_CLCD - trace

This section describes the trace sources.

CHECKPOINT_MESSAGE

Report error messages from the checkpointing process. Fields:

message string
: Message contents.

CHECKPOINT_RESTORE_END

Checkpoint restore completed.

CHECKPOINT_RESTORE_START

Checkpoint restore about to start.

CHECKPOINT_SAVE_END

Checkpoint save completed.

CHECKPOINT_SAVE_START

Checkpoint save about to start.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.96 PL310_L2CC - trace

This section describes the trace sources.

CACHE_READ_HIT

A line read hit occurred. Fields:

addr `unsigned int`
: Address of transaction

cache_line_index `unsigned int`
: Cache line index.

ns `bool`
: Non-secure.

CACHE_READ_MISS

A line read miss occurred. Fields:

addr `unsigned int`
: Address of transaction

ns `bool`
: Non-secure.

CACHE_WRITE_HIT

A line write hit occurred. Fields:

addr `unsigned int`
: Address of transaction

cache_line_index `unsigned int`
: Cache line index.

ns `bool`
: Non-secure.

CACHE_WRITE_MISS

A line write miss occurred. Fields:

addr `unsigned int`
: Address of transaction

ns `bool`
: Non-secure.

cache_line_eviction

The cache is starting to evict a line to downstream. Fields:

addr `unsigned int`
: Address.

cache_line_index `unsigned int`
: Cache line index

data `unsigned int`
: Data being evicted.

memory_attributes `enum`
: Memory attributes

ns `bool`
: Non-secure.

cache_line_refill

The cache refilled a line from downstream. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

addr unsigned int
: Address.

cache_line_index unsigned int
: Cache line index.

data unsigned int
: Data or empty on error.

memory_attributes enum
: Memory attributes

ns bool
: Non-secure.

cache_line_refill_start

The cache is starting to refill a line from downstream. Fields:

addr unsigned int
: Address.

cache_line_index unsigned int
: Cache line index.

memory_attributes enum
: Memory attributes

ns bool
: Non-secure.

interrupt_status

The raw interrupt status. Fields:

mask_of_new_bit unsigned int
: A mask of which bit was changed.

new_value unsigned int
: New interrupt status value.

register_read

A register read occurred. Fields:

data unsigned int
: Data read

name string
: The register name.

ns bool
: Access is non-secure

offset unsigned int
: Offset of register in file.

ok bool
: True if a valid register.

register_write

A register write occurred. Fields:

data unsigned int
: Data written

name string
: The register name.

ns bool
: Access is non-secure

offset unsigned int
: Offset of register in file.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ok `bool`
True if ok.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.97 PL330_DMAC - trace

This section describes the trace sources.

channel_read

A completed channel read. Fields:

acache `unsigned int`
: The ACACHE value expanded to four bits.

address `unsigned int`
: Address of transaction.

aprot `unsigned int`
: The APROT value.

channel_number `unsigned int`
: Channel making transaction.

data `unsigned int`
: The data.

error `bool`
: The transaction produced an error.

width_in_bytes `unsigned int`
: Width in bytes.

channel_write

A completed channel write. Fields:

acache `unsigned int`
: The ACACHE value expanded to four bits.

address `unsigned int`
: Address of transaction.

aprot `unsigned int`
: The APROT value.

channel_number `unsigned int`
: Channel making transaction.

data `unsigned int`
: The data.

error `bool`
: The transaction produced an error.

width_in_bytes `unsigned int`
: Width in bytes.

exec_channel_instruction

About to execute a channel instruction. Fields:

channel `unsigned int`
: Channel about to execute instruction.

opcode `enum`
: Opcode about to be executed.

exec_manager_instruction

About to execute a manager instruction. Fields:

opcode `enum`
: Opcode about to be executed.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

fetch_instruction_byte

A byte was fetched for an instruction, this could include extra arguments fetched for multi-byte instructions. Fields:

address unsigned int
: Address of access.

data unsigned int
: Data returned if OK.

ns bool
: NS state of access.

ok bool
: Was the access OK?

inst_dmaend

DMAEND instruction executed by manager thread.

inst_dmago

DMAGO instruction. Fields:

channel_number unsigned int
: Channel issuing instruction.

cns bool
: The thread in non-secure.

pc unsigned int
: Program counter.

inst_dmago_no_go

DMAGO instruction failed. Fields:

channel_number unsigned int
: Channel issuing instruction.

why enum
: Why failed

inst_dmakill

DMAKILL instruction executed by manager thread.

inst_dmald

DMALD instruction. Fields:

channel_number unsigned int
: Channel issuing instruction.

increment_addr bool
: Whether to increment the address or not.

number_of_beats unsigned int
: Number of beats to transfer.

source_address unsigned int
: Source address.

transfer_size_in_bytes unsigned int
: Transfer size in bytes.

inst_dmalp

DMALP instruction executed by channel, loads 'iterations' into the loop counter. Fields:

channel_number unsigned int
: Channel issuing instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

iterations unsigned int
: The value to load into the loop counter.

loop_counter unsigned int
: Which loop counter.

inst_dmamov

DMAMOV instruction. Fields:

channel_number unsigned int
: Channel issuing instruction.

data unsigned int
: Data moved.

register enum
: Register being moved to.

inst_dmanop

DMANOP instruction executed by channel. Fields:

channel_number unsigned int
: Channel issuing instruction.

inst_dmanop_manager

DMANOP instruction executed by manager.

inst_dmasev

DMASEV instruction. Fields:

channel_number unsigned int
: Channel issuing instruction.

irq_number unsigned int
: IRQ number.

inst_dmasev_manager

DMASEV instruction executed by manager. Fields:

irq_number unsigned int
: IRQ number.

inst_dmast

DMAST instruction. Fields:

channel_number unsigned int
: Channel issuing instruction.

destination_address unsigned int
: Destination address.

increment_addr bool
: Whether to increment the address or not.

number_of_beats unsigned int
: Number of beats to transfer.

transfer_size_in_bytes unsigned int
: Transfer size in bytes.

inst_dmawfe

DMAWFE instruction. Fields:

channel_number unsigned int
: Channel issuing instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

pending_event_found unsigned int
The DMAWFE doesn't wait as event is already signaled.

inst_dmawfe_manager

DMAWFE instruction executed by manager.

inst_instruction_not_implemented

Instruction is not implemented by the model. Fields:

which enum
Which instruction is not modelled.

mgr_evnt_err

Manager thread in the Non-secure state attempted to execute either a DMAWFE to wait for a secure event or DMASEV to create a secure event or secure interrupt. Fields:

irq_number unsigned int
Event-interrupt number.
opcode enum
Opcode of the attempted instruction.

register_read

A register read occurred. Fields:

data unsigned int
The data returned if not an error.
error bool
An error is being returned.
ns bool
The security state of the access.
offset enum
The offset of the register access.

register_write

A register write occurred. Fields:

data unsigned int
The write data set if not an error.
error bool
An error is being returned.
ns bool
The security state of the access.
offset enum
The offset of the register access.

start_channel_read

Start of a channel read. Fields:

acache unsigned int
The ACACHE value expanded to four bits.
address unsigned int
Address of transaction.
aprot unsigned int
The APROT value.
channel_number unsigned int
Channel making transaction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

width_in_bytes unsigned int
: Width in bytes.

start_channel_write

Start of a channel write. Fields:

acache unsigned int
: The ACACHE value expanded to four bits.

address unsigned int
: Address of transaction.

aprot unsigned int
: The APROT value.

channel_number unsigned int
: Channel making transaction.

width_in_bytes unsigned int
: Width in bytes.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.98 PPUv0 - trace

This section describes the trace sources.

PPU_AccessToUnimplementedRegister

Fields:

ADDR `unsigned int`
: Address of the accessed register.

READ `bool`
: Transaction type is read or write.

PPU_ReadToWriteOnlyRegister

Fields:

ADDR `unsigned int`
: Address of the accessed register.

PPU_RegRead

Fields:

REG_NAME `string`
: The name of the register read.

REG_OFFSET `unsigned int`
: The address of the register read.

VALUE `unsigned int`
: Read Value from the register.

PPU_RegWrite

Fields:

REG_NAME `string`
: The name of the register updated.

REG_OFFSET `unsigned int`
: The address of the register updated.

UPDATED_VALUE `unsigned int`
: New Value in the register.

VALUE `unsigned int`
: Old Value in the register.

PPU_WriteToReadOnlyRegister

Fields:

ADDR `unsigned int`
: Address of the accessed register.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.99 PPUv1 - trace

This section describes the trace sources.

PPU_AccessToUnimplementedRegister

Fields:

ADDR unsigned int
: Address of the accessed register.

READ bool
: Transaction type is read or write.

PPU_ReadToWriteOnlyRegister

Fields:

ADDR unsigned int
: Address of the accessed register.

PPU_RegRead

Fields:

REG_NAME string
: The name of the register read.

REG_OFFSET unsigned int
: The address of the register read.

VALUE unsigned int
: Read Value from the register.

PPU_RegWrite

Fields:

REG_NAME string
: The name of the register updated.

REG_OFFSET unsigned int
: The address of the register updated.

UPDATED_VALUE unsigned int
: New Value in the register.

VALUE unsigned int
: Old Value in the register.

PPU_WriteToReadOnlyRegister

Fields:

ADDR unsigned int
: Address of the accessed register.

key_registers_value_when_power_state_change

trace the key register value when power state may change. Fields:

PPU_DISR unsigned int
PPU_IESR unsigned int
PPU_ISR unsigned int
PPU_PWPR unsigned int
PPU_PWSR unsigned int
Source string

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

lock_status

The current PPU OFF lock status is %{STATUS}. Fields:

STATUS enum
: The status of PPU OFF lock

pactive_port_status

Detected PACTIVE port signal change %{PORT_STATUS} with value %{SIG_VALUE:x}. Fields:

PORT_STATUS enum
: The status of PPU PACTIVE port

SIG_VALUE unsigned int
: The value of PACTIVE signal

receive_wakerequest

trace the wakerequest event. Fields:

Wakerequest string
: PPU is receiving Wakerequest

transition_status

The current PPU transition status is %{STATUS}. Fields:

STATUS enum
: The status of PPU transition

wakerequest_failed

The wakerequest failed and PPU_ISR is %{PPU_ISR}, PPU_IESR is %{PPU_IESR}. Fields:

PPU_IESR unsigned int
: The value of PPU_IESR when wakerequest is failed

PPU_ISR unsigned int
: The value of PPU_ISR when wakerequest is failed

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.100 PVBus2AMBAPVACE - trace

This section describes the trace sources.

ACE_DVM

ACE dvm trace message. Fields:

MSG `string`
: DVM Message

ACE_SNOOP

ACE snoop trace data. Fields:

ADDR_END_INCL `unsigned int`
: Physical address of access - end inclusive

ADDR_START `unsigned int`
: Physical address of access - start

IS_DATATRANS `bool`
: Is a data transfer?

IS_ERROR `bool`
: Is an error?

IS_PASSDIRTY `bool`
: Is pass dirty?

IS_SHARED `bool`
: Is shared?

IS_WASUNIQUE `bool`
: Was unique?

NS `unsigned int`
: Is a non secure transaction?

Snoop `enum`
: Encodes the ACE operation: 0 ReadNoSnoop, 1 ReadOnce, 2 ReadClean, 3 ReadShared, 4 ReadNotSharedDirty, 5 ReadUnique, 6 CleanShared, 7 CleanInvalid, 8 MakeInvalid, 9 CleanUnique ,10 MakeUnique, 11 WriteNoSnoop, 12 PseudoWriteOnce, 13 WriteUnique, 14 WriteLineUnique, 15 WriteBack, 16 WriteClean, 17 Evict, 18 Default, 19 MemoryBarrier, 20 SynchronisationBarrier, 21 NoOperation, 22 DebugRead, 23 DebugWrite

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.101 PVBusBridge - trace

This section describes the trace sources.

BRIDGE_DMI

DMI information from PVBus to AMBA-PV bridge. Fields:

ADDR `unsigned int`
: Address of the operation

Access `string`
: R = Read, W = Write, r = Prefetch only read, w = Prefetch only write, I = Invalidate

Op `string`
: D = Downstream call, N = New range received, I = Invalidate from downstream, M = invalidate from upstream monitor

R_END `unsigned int`
: Range end

R_PTR `unsigned int`
: Range pointer

R_START `unsigned int`
: Range start

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.102 PVBusExclusiveMonitor - trace

This section describes the trace sources.

CHECKPOINT_MESSAGE

Report error messages from the checkpointing process. Fields:

message string
: Message contents.

CHECKPOINT_RESTORE_END

Checkpoint restore completed.

CHECKPOINT_RESTORE_START

Checkpoint restore about to start.

CHECKPOINT_SAVE_END

Checkpoint save completed.

CHECKPOINT_SAVE_START

Checkpoint save about to start.

GLOBAL_MONITOR

Global monitor activity. Fields:

MASTER_ID unsigned int
: Indicates the master associated with this monitor

NS enum
: Secure state of the region monitored

PADDR unsigned int
: Physical address of the region monitored

SIZE unsigned int
: Size of the region monitored

STATE enum
: State of the monitor (Open/Exclusive)

TRIGGER enum
: Reason the monitor changed state

WIDTH unsigned int
: Width in bytes of the access which created the exclusive region

MONITOR_TRANSACTION

Transactions observed by the monitor. Fields:

DATA unsigned int
: The data transfered by the transaction

MASTER_ID unsigned int
: Identifies the master of this transaction

NS enum
: Secure state of the transaction

PADDR unsigned int
: Physical address of the transaction

RESPONSE enum
: The transaction result

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

TYPE `enum`
: Type of transaction

WIDTH `unsigned int`
: Width in bytes of the access

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.103 PVBusGICv3Comms - trace

This section describes the trace sources.

ArchMsg.Warning.GICv3CommsPVBus_PacketTooBig

Fields:

DATA_0 unsigned int
: First byte of the packet (normally data type).

INDEX unsigned int
: Index of GICv3Comms connection that the message should have been sent to (based on AXI master ID).

LENGTH unsigned int
: Number bytes in data packet.

ArchMsg.Warning.GICv3CommsPVBus_UnknownDestination

Fields:

DATA unsigned int
: The message data.

DATA_0 unsigned int
: First byte of the packet (normally data type).

INDEX unsigned int
: Index of GICv3Comms connection that the message should have been sent to (based on AXI master ID).

LENGTH unsigned int
: Number bytes in data packet.

GICv3CommsPVBus_FailedMessage

Processed message from connected GICv3Comms port and sent out, but PVBus transaction failed.
Fields:

DATA unsigned int
: The message data.

INDEX unsigned int
: Index of GICv3Comms connection that message came in on.

LENGTH unsigned int
: Message length in bytes.

GICv3CommsPVBus_ProcessedMessage

Processed message from connected GICv3Comms port. Fields:

DATA unsigned int
: The message data.

INDEX unsigned int
: Index of GICv3Comms connection that message came in on.

LENGTH unsigned int
: Message length in bytes.

GICv3CommsPVBus_ReceiveAXIMasterID

Fields:

INDEX unsigned int
: Index of GICv3Comms connection that message came in on.

MASTER_ID unsigned int
: Master ID value recieved.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

GICv3CommsPVBus_SentViaGICv3Comms

Packet received via PVBus and sent via GICv3 Comms to connected distributor/CPU. Fields:

DATA `unsigned int`
: The message data.

INDEX `unsigned int`
: Index of GICv3Comms connection that the message should have been sent to (based on AXI master ID).

LENGTH `unsigned int`
: Number bytes in data packet.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.104 PVBusLogger - trace

This section describes the trace sources.

READ_ACCESS

Trace access at this point in the PVBus hierarchy. Fields:

ACCESS_SIZE unsigned int
: Log2 of access size (i.e. 0=byte, 1=halfword, 2=word, 3=doubleword ...)

ACE enum
: Encodes the ACE operation

ATTR unsigned int
: Transaction Attributes: [12] Instruction fetch [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

DATA unsigned int
: The data transferred

EXTENDED_ID unsigned int
: The Extended ID

LATENCY unsigned int
: Time downstream (ticks)

MASTER_ID unsigned int
: The AXI Master ID

NUMBER_OF_BEATS unsigned int
: The number of data transfers (beats) in this burst

PADDR unsigned int
: Physical address of access

RESPONSE enum
: Whether the transaction was successful, or an error occurred

USER_FLAGS unsigned int
: Core specific additional signals

READ_ACCESS_START

Trace access at this point in the PVBus hierarchy. Fields:

ACCESS_SIZE unsigned int
: Log2 of access size (i.e. 0=byte, 1=halfword, 2=word, 3=doubleword ...)

ACE enum
: Encodes the ACE operation

ATTR unsigned int
: Transaction Attributes: [12] Instruction fetch [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

DATA unsigned int
: The data transferred

EXTENDED_ID unsigned int
: The Extended ID

MASTER_ID unsigned int
: The AXI Master ID

NUMBER_OF_BEATS unsigned int
: The number of data transfers (beats) in this burst

PADDR unsigned int
: Physical address of access

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

USER_FLAGS unsigned int
: Core specific additional signals

WRITE_ACCESS

Trace access at this point in the PVBus hierarchy. Fields:

ACCESS_SIZE unsigned int
: Log2 of access size (i.e. 0=byte, 1=halfword, 2=word, 3=doubleword ...)

ACE enum
: Encodes the ACE operation

ATTR unsigned int
: Transaction Attributes: [12] Instruction fetch [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

DATA unsigned int
: The data transferred

EXTENDED_ID unsigned int
: The Extended ID

LATENCY unsigned int
: Time downstream (ticks)

MASTER_ID unsigned int
: The AXI Master ID

NUMBER_OF_BEATS unsigned int
: The number of data transfers (beats) in this burst

PADDR unsigned int
: Physical address of access

RESPONSE enum
: Whether the transaction was successful, or an error occurred

USER_FLAGS unsigned int
: Core specific additional signals

WRITE_ACCESS_START

Trace access at this point in the PVBus hierarchy. Fields:

ACCESS_SIZE unsigned int
: Log2 of access size (i.e. 0=byte, 1=halfword, 2=word, 3=doubleword ...)

ACE enum
: Encodes the ACE operation

ATTR unsigned int
: Transaction Attributes: [12] Instruction fetch [11] Non-secure, [10] Privileged, [9:8] shareability domain (0=nsh, 1=ish, 2=osh, 3=system), [7:4] outer memory attributes, [3:0] inner memory attributes ([7]/[3] Allocate (Other Allocate when read), [6]/[2] Allocate (Other Allocate when write), [5]/[1] Modifiable, [4]/[0] Bufferable). All other bits are implementation defined

DATA unsigned int
: The data transferred

EXTENDED_ID unsigned int
: The Extended ID

MASTER_ID unsigned int
: The AXI Master ID

NUMBER_OF_BEATS unsigned int
: The number of data transfers (beats) in this burst

PADDR unsigned int
: Physical address of access

USER_FLAGS unsigned int
: Core specific additional signals

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.105 PVBusMapper - trace

This section describes the trace sources.

ArchMsg.Error.BusActiveDuringReset

Transactions recieved at the bus slave port whilst reset was asserted.

ArchMsg.Warning.BusDeadlockDetected

A potential bus deadlock has been detected. Fields:

INTERCONNECT_RESET bool
: Downstream bus interconnect reset

INTERMEDIARY_RESET bool
: Bus mapper intermediary reset

SLAVE_RESET bool
: Downstream bus slave reset

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.106 PVBusMaster - trace

This section describes the trace sources.

ArchMsg.Warning.BusDeadlockDetected

A potential bus deadlock has been detected. Fields:

INTERCONNECT_RESET bool
: Bus interconnect reset

MASTER_RESET bool
: Bus master reset

SLAVE_RESET bool
: Bus slave reset

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.107 PVBusSlave - trace

This section describes the trace sources.

ATOMIC_SLAVE_ACCESS

Information on the atomic operation at slave side. Fields:

ACCESS_RESULT enum
: Result of atomic access

COMPARE_VALUE unsigned int
: Value to be compared with memory by CAS

LOAD_VALUE unsigned int
: Value to be loaded from memory as a result of atomic operation

MASTER unsigned int
: ID of master which initiated the transaction

NS bool
: Transaction is Non-Secure

OPERAND_VALUE unsigned int
: Incoming operand value provided by core

OPERATION unsigned int
: Type of atomic operation

PADDR unsigned int
: Physical address

STORE_VALUE unsigned int
: Value to be stored to memory as a result of atomic operation

ArchMsg.Warning.uninitialized_data_read

Unwritten data read warning for ACCESSMODE_TRACKED_MEMORY. Fields:

MASK unsigned int
: Set bits indicate uninitialized data in bits [63:0]

MASK2 unsigned int
: Set bits indicate uninitialized data in bits [127:64]

PADDR unsigned int
: Physical address

WIDTH unsigned int
: Bit width of access

ArchMsg.Warning.unmonitored_exclusive_access

Exclusive access to address not covered by an exclusive monitor. Fields:

DIRECTION enum
: Access direction (read/write)

NS bool
: Transaction is Non-Secure

PADDR unsigned int
: Physical address

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.108 PVCache - trace

This section describes the trace sources.

ALLOC_LINEFILL

The system allocated a complete line fill into the RAMs. This happens when the cache has to read the data for a line fill from potentially upstream or downstream. Fields:

ENTRY_INDEX `unsigned int`
: The entry index the line has been loaded into.

MASTER_ID `unsigned int`
: The master ID for the associated transaction.

MEMORY_ATTRIBUTES `unsigned int`
: The memory attributes used for the allocation. bits[3:0] are the inner ACACHE, bits[7:4] are the outer ACACHE, bits[9:8] are ADOMAIN, bits[12:10] are APROT, the rest are undocumented.

NS_ADDR `unsigned int`
: The security world and address of the first byte of the line, the MSB encodes the NS bit.

ALLOC_WRITE

The system allocated a complete line write into the RAMs. This happens when the cache receives a complete cache line write. It need not read from anywhere. Fields:

ENTRY_INDEX `unsigned int`
: The entry index the line has been written into.

MEMORY_ATTRIBUTES `unsigned int`
: The memory attributes used for the allocation. bits[3:0] are the inner ACACHE, bits[7:4] are the outer ACACHE, bits[9:8] are ADOMAIN, bits[12:10] are APROT, the rest are undocumented.

NS_ADDR `unsigned int`
: The security world and address of the first byte of the line, the MSB encodes the NS bit.

ATOMIC_SLAVE_ACCESS

Information on the atomic operation at slave side. Fields:

ACCESS_RESULT `enum`
: Result of atomic access

COMPARE_VALUE `unsigned int`
: Value to be compared with memory by CAS

LOAD_VALUE `unsigned int`
: Value to be loaded from memory as a result of atomic operation

MASTER `unsigned int`
: ID of master which initiated the transaction

NS `bool`
: Transaction is Non-Secure

OPERAND_VALUE `unsigned int`
: Incoming operand value provided by core

OPERATION `enum`
: Type of atomic operation

PADDR `unsigned int`
: Physical address

STORE_VALUE `unsigned int`
: Value to be stored to memory as a result of atomic operation

ArchMsg.Error.cached_attributes_mismatch

Attributes differ between page and line in cache from same page. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

LINEATTR unsigned int
: Attributes of the cache allocated lines

LINES unsigned int
: Number of cache lines already allocated

NS_ADDR unsigned int
: The security world and address of the first byte of the page, the MSB encodes the NS bit.

TXATTR unsigned int
: Attributes of the transaction

ArchMsg.Error.cached_attributes_mismatch#lineinfo

Information about allocated lines in page. Fields:

PADDR unsigned int
: address of this cache line

TAG unsigned int
: tag data of this cache line

ArchMsg.Warning.ns_s_dirty_hit

Cache hit at S and NS versions of the same physical address. Fields:

INDEX0 unsigned int
: The cache line index of the affected cache line

INDEX1 unsigned int
: The cache line index of the other regime cache line

PADDR unsigned int
: The address of the affected cache line

TAG0 unsigned int
: The tag data of the affected cache line

TAG1 unsigned int
: The tag data of the other regime cache line

CACHE_READ_HIT

Read access cache hit. Note that this trace source causes a large slowdown in the simulation. Fields:

IS_PRELOAD enum
: Is the access a preload

IS_SHARED enum
: Is the access shared

LATENCY unsigned int
: Cache latency in ticks per access

MASTER_ID unsigned int
: Indicates the master associated with this monitor

CACHE_READ_MISS

Read access cache miss and cache miss latency. Fields:

IS_PRELOAD enum
: Is the access a preload

IS_SHARED enum
: Is the access shared

LATENCY unsigned int
: Cache miss latency in ticks per access

MASTER_ID unsigned int
: Indicates the master associated with this monitor

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CACHE_WRITE_HIT

Write access cache hit. Note that this trace source causes a large slowdown in the simulation. Fields:

IS_PRELOAD enum
: Is the access a preload

IS_SHARED enum
: Is the access shared

LATENCY unsigned int
: Cache latency in ticks per access

MASTER_ID unsigned int
: Indicates the master associated with this monitor

CACHE_WRITE_MISS

Write access cache miss and cache miss latency. Fields:

IS_PRELOAD enum
: Is the access a preload

IS_SHARED enum
: Is the access shared

LATENCY unsigned int
: Cache miss latency in ticks per access

MASTER_ID unsigned int
: Indicates the master associated with this monitor

ENTRY_BECOMES_INVALID

An entry is now invalid. The cause could be any reason. Fields:

ENTRY_INDEX unsigned int
: The entry index that has become invalid.

ENTRY_SET_DIRTY

Entry change, clean/dirty. Fields:

ENTRY_INDEX unsigned int
: The entry index that has become clean or dirty.

STATUS unsigned int
: True if the line was made dirty

ERROR_MIXED_ATTRIBUTES_LINE

Mismatched line and transaction attributes. Fields:

ENTRY_INDEX unsigned int
: The entry index that has collided.

NS_ADDR unsigned int
: The security world and address of the first byte of the line, the MSB encodes the NS bit.

ERROR_MIXED_ATTRIBUTES_PAGE

Attributes differ between page and lines in cache from the same page. Fields:

NS_ADDR unsigned int
: The security world and address of the first byte of the page, the MSB encodes the NS bit.

EVICTION

A line was evicted from the cache. Fields:

ADDRESS unsigned int
: Base address of the victim cache line

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CLEAN bool
: If true the cache retained the cache line after writing the dirty data

ENTRY_INDEX unsigned int
: The entry index of the victim

NON_SECURE bool
: If true the victim was NonSecure

IMPRECISE_ABORT

An operation produced an imprecise abort. Usually this is a WriteBack/WriteClean operation receiving an error response. Fields:

ADDRESS unsigned int
: Address of the transaction that imprecise aborted.

DECERR bool
: True if this is a DECERR response.

DOWNSTREAM_PORT_INDEX unsigned int
: The downstream port index that received the imprecise abort

NS bool
: The NS state of the transaction (true is non-secure)

MAINTENANCE_CLEAN_SET_WAY

A cache maintenance by set-way operation occurred, possibly the 'clean' part of a clean-and-invalidate operation. Fields:

ENTRY_INDEX unsigned int
: The entry index the set-way operation has mapped to.

IS_NON_SECURE bool
: If the invalidate operation is a non-secure (and so only affects non-secure lines), or secure (and so affects both).

STATE enum
: Line transformation incurred

MAINTENANCE_INVALIDATE_ALL

A cache maintenance invalidate-all operation occurred. Fields:

IS_NON_SECURE bool
: If the invalidate operation is a non-secure (and so only affects non-secure lines), or secure (and so affects both).

MAINTENANCE_INV_SET_WAY

A cache maintenance by set-way operation occurred, possibly the 'invalidate' part of a clean-and-invalidate operation. Fields:

ENTRY_INDEX unsigned int
: The entry index the set-way operation has mapped to.

IS_NON_SECURE bool
: If the invalidate operation is a non-secure (and so only affects non-secure lines), or secure (and so affects both).

STATE enum
: Line transformation incurred

change_dvm_disabled_ports

This reports a bitmap of which ports are disabled to receive DVM messages. Fields:

new_bitmap unsigned int
: The new bitmap of disabled ports

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

new_bitmap_list unsigned int
: A list of port numbers that were disabled.

old_bitmap unsigned int
: The old bitmap of disabled ports

old_bitmap_list unsigned int
: A list of port numbers that were disabled.

change_upstream_disabled_ports

This reports a bitmap of which upstream ports of the cache are _disabled_ and so we won't sends snoop requests to them. Fields:

new_bitmap unsigned int
: The bitmap of the first 32 upstream ports that are disabled.

new_bitmap_list unsigned int
: A list of port numbers that were disabled.

old_bitmap unsigned int
: The bitmap of the first 32 upstream ports that are disabled.

old_bitmap_list unsigned int
: A list of port numbers that were disabled.

entry_after_far_atomic

A cache line has had a far atomic applied to it. Fields:

atomic_operation enum
: The operation that operated on this line.

be bool
: The far atomic's big-endian flag.

data unsigned int
: The data in the cache line.

entry_begin_address unsigned int
: The start address of the entry.

entry_index unsigned int
: The entry index in the cache.

incoming_data unsigned int
: The incoming data from the far atomic. For CAS operations then this is compare-value and then the replacement-value

master_id unsigned int
: The master ID for the associated transaction

ns enum
: The security state of the line

prior_data unsigned int
: The data that will be returned as the 'prior' data of the far atomic.

trans_begin_address unsigned int
: The start address of the transaction.

entry_after_read

A cache line has been read. Fields:

ace_operation enum
: The ACE operation that operated on this line.

data unsigned int
: The data in the cache line.

dirty bool
: Is the line ACE-dirty.

entry_begin_address unsigned int
: The start address of the entry.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

entry_index unsigned int
: The entry index in the cache.

ns enum
: The security state of the line

snoop bool
: This transaction is a snoop transaction

trans_begin_address unsigned int
: The start address of the transaction.

unique bool
: Is the line ACE-unique.

entry_after_write

A cache line has been written. Fields:

ace_operation enum
: The ACE operation that operated on this line.

data unsigned int
: The data in the cache line.

entry_begin_address unsigned int
: The start address of the entry.

entry_index unsigned int
: The entry index in the cache.

ns enum
: The security state of the line

trans_begin_address unsigned int
: The start address of the transaction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.109 PVWriteBuffer - trace

This section describes the trace sources.

ATOMIC_SLAVE_ACCESS

Information on the atomic operation at slave side. Fields:

ACCESS_RESULT enum
: Result of atomic access

COMPARE_VALUE unsigned int
: Value to be compared with memory by CAS

LOAD_VALUE unsigned int
: Value to be loaded from memory as a result of atomic operation

MASTER unsigned int
: ID of master which initiated the transaction

NS bool
: Transaction is Non-Secure

OPERAND_VALUE unsigned int
: Incoming operand value provided by core

OPERATION unsigned int
: Type of atomic operation

PADDR unsigned int
: Physical address

STORE_VALUE unsigned int
: Value to be stored to memory as a result of atomic operation

WRITE_BUFFER_CONTENTS

Contents of a Line in WriteBuffer before/after updating them. Fields:

ADDRESS unsigned int
: Address

BEFORE_UPDATE bool
: Before update

DATA unsigned int
: Data content

LINE_INDEX unsigned int
: Line index

METADATA unsigned int
: MetaData value

NS bool
: Normal(true), Secure(false)

WRITE_BUFFER_HIT

An access hits in WriteBuffer. Fields:

ADDRESS unsigned int
: Address

IS_IMMEDIATELY_SERVED_BY_WB bool
: true: This access is serviced in WriteBuffer? false: Sent downstream

IS_READ bool
: Is Read

NS bool
: Normal(true), Secure(false)

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

WRITE_BUFFER_READ_DATA_DOWNSTREAM

Read one line of data from downstream. Fields:

ADDRESS unsigned int
: Address

DATA unsigned int
: Data content

NS bool
: Normal(true), Secure(false)

WRITE_BUFFER_READ_METADATA_DOWNSTREAM

Read one MetaDataElement_t from downstream. Fields:

ADDRESS unsigned int
: Address

METADATA unsigned int
: MetaData read from downstream

NS bool
: Normal(true), Secure(false)

WRITE_BUFFER_WARNING_RESET_WHEN_DIRTY

Reset received whilst dirty lines exist in the write buffer.

WRITE_BUFFER_WARNING_RESET_WHEN_IN_TRANSACTIONAL_STATE

Reset received whilst the write buffer constains TME Transactional state.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.110 RAMDevice - trace

This section describes the trace sources.

CHECKPOINT_MESSAGE

Report error messages from the checkpointing process. Fields:

message `string`
: Message contents.

CHECKPOINT_RESTORE_END

Checkpoint restore completed.

CHECKPOINT_RESTORE_START

Checkpoint restore about to start.

CHECKPOINT_SAVE_END

Checkpoint save completed.

CHECKPOINT_SAVE_START

Checkpoint save about to start.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.111 SMMUv3AEM - trace

This section describes the trace sources.

ArchMsg.Error.error

These messages are about activity occurring on the SMMU that is considered an error. Messages will only come out here if parameter all_error_messages_through_trace is true. Fields:

output string
: The stream output

ArchMsg.Error.fetch_from_memory_type_not_supporting_httu

A descriptor fetch from an HTTU-enabled translation regime to an unsupported memory type was made. Whilst the fetch itself may succeed, if an update to the descriptor was attempted then it would fail. Fields:

address unsigned int
: The address of the descriptor fetch.

desc_inner enum
: Inner cacheability of descriptor

desc_outer enum
: Outer cacheability of descriptor

desc_prot enum
: The instruction/data, bus-ns/s and privileged/user marking

desc_sh enum
: Shareability of descriptor

ssd_ns bool
: Is the Security State Determination of the transaction non-secure.

stage unsigned int
: The stage at which we had a problem.

streamid unsigned int
: The StreamID of the transaction

substreamid unsigned int
: The SubstreamID of the transaction (or ~0u if no substreamid)

trans_id unsigned int
: Transaction id.

ArchMsg.Error.ns_cmdq

There is an error associated with the command queue. Fields:

cmd_id unsigned int
: Command ID of the command in error.

cons unsigned int
: CONS of the command.

what string
: What is wrong.

ArchMsg.Error.priq_streamid_truncated

The SMMU received a PCIe PRI request with a StreamID) that was larger than that which the SMMU has been configured for. The StreamID that appears in the PRIQ entry will be truncated. Fields:

actual_streamid unsigned int
: The actual StreamID that this request has.

sidsize unsigned int
: The bit width of the SMMU for StreamIDs, as indicated by SMMU_IDR1.SIDSIZE.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

trans_id unsigned int
: The transaction ID that identifies this request.

trunc_streamid unsigned int
: The truncated StreamID that will appear in the PRIQ entry.

ArchMsg.Error.priq_substreamid_truncated

The SMMU received a PCIe PRI request with a PASID prefix (SubstreamID) that was larger than that which the SMMU has been configured for. The SubstreamID that appears in the PRIQ entry will be truncated. Fields:

actual_substreamid unsigned int
: The actual SubstreamID that this request has.

ssidsize unsigned int
: The bit width of the SMMU for SubstreamIDs, as indicated by SMMU_IDR1.SSIDSIZE.

trans_id unsigned int
: The transaction ID that identifies this request.

trunc_substreamid unsigned int
: The truncated SubstreamID that will appear in the PRIQ entry.

ArchMsg.Error.s_cmdq

There is an error associated with the command queue. Fields:

cmd_id unsigned int
: Command ID of the command in error.

cons unsigned int
: CONS of the command.

what string
: What is wrong.

ArchMsg.Error.tlb_entries_overlap

A TLB entry was attempted to be inserted into the TLB and was determined that it overlaps an existing entry. This check is not perfect but will catch simple errors. Fields:

do_f_tlb_conflict bool
: Chosen to perform an F_TLB_CONFLICT.

end_address_of_new_entry unsigned int
: End address of new entry.

end_address_of_old_entry unsigned int
: End address of old entry.

index_of_new_entry unsigned int
: Index of new entry.

index_of_old_entry unsigned int
: Index of old entry.

start_address_of_new_entry unsigned int
: Start address of new entry.

start_address_of_old_entry unsigned int
: Start address of old entry.

why enum
: Why it is an error for these two entries to overlap.

ArchMsg.Error.tlb_entry_not_invalidated_due_to_ril

An entry in the cache was not invalidated even though in the right address range because of the RIL fields of the command the entry do not match. Fields:

cmd_num unsigned int
: The NUM field of the RIL part of the command.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

cmd_ril_tg enum
: The RIL_TG field of the RIL part of the command.

cmd_ril_ttl unsigned int
: The TTL field of the RIL part of the command, zero means any level.

cmd_scale unsigned int
: The SCALE field of the RIL part of the command.

entry_id unsigned int
: The entry id that is being invalidated.

tlb_entry string
: The TLB entry.

ArchMsg.Error.vatos_sel_vmid_out_of_range

The SMMU_(S_)VATOS_SEL.VMID field was programmed with a VMID that was too wide for this implementation (SMMU_IDR0.VMID16 == 0). Fields:

ssd_ns bool
: The security state of the VATOS interface

vmid unsigned int
: The VMID programmed.

ArchMsg.Info.info

These are information messages about what is happening in the SMMU. Fields:

output string
: The stream output

ArchMsg.Warning.CMD_RESUME_no_transaction_resumed

A CMD_RESUME was issued that matched no transaction. Fields:

stag unsigned int
: STAG in the CMD_RESUME.

streamid unsigned int
: StreamID in the CMD_RESUME.

streamid_ns bool
: The StreamID was for the non-secure world.

ArchMsg.Warning.atc_inv_strange

Something was odd about the CMD_ATC_INV. Fields:

cmd_id unsigned int
: Command id

why string
: Why the CMD_ATC_INV was strange

ArchMsg.Warning.bad_conf_reset_of_SMMU_GBPA_ABORT

The signal was not driven at reset by the external system, but was driven after we first started using the default value we had assumed. The external system, if it is going to drive the signal, must do so at first reset so that both sides of the connection can agree on the value of this signal. Fields:

default_value bool
: The default value of the signal we have been assuming.

value bool
: The value of the signal.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.bad_conf_reset_of_SMMU_S_GBPA_ABORT

The signal was not driven at reset by the external system, but was driven after we first started using the default value we had assumed. The external system, if it is going to drive the signal, must do so at first reset so that both sides of the connection can agree on the value of this signal. Fields:

default_value bool
: The default value of the signal we have been assuming.

value bool
: The value of the signal.

ArchMsg.Warning.bad_conf_system_supports_btm

The signal was not driven at reset by the external system, but was driven after we first started using the default value we had assumed. The external system, if it is going to drive the signal, must do so at first reset so that both sides of the connection can agree on the value of this signal. Fields:

default_value bool
: The default value of the signal we have been assuming.

value bool
: The value of the signal.

ArchMsg.Warning.bad_conf_system_supports_cohacc

The signal was not driven at reset by the external system, but was driven after we first started using the default value we had assumed. The external system, if it is going to drive the signal, must do so at first reset so that both sides of the connection can agree on the value of this signal. Fields:

default_value bool
: The default value of the signal we have been assuming.

value bool
: The value of the signal.

ArchMsg.Warning.bad_conf_system_supports_httu

The signal was not driven at reset by the external system, but was driven after we first started using the default value we had assumed. The external system, if it is going to drive the signal, must do so at first reset so that both sides of the connection can agree on the value of this signal. Fields:

default_value bool
: The default value of the signal we have been assuming.

value bool
: The value of the signal.

ArchMsg.Warning.bad_conf_system_supports_sev

The signal was not driven at reset by the external system, but was driven after we first started using the default value we had assumed. The external system, if it is going to drive the signal, must do so at first reset so that both sides of the connection can agree on the value of this signal. Fields:

default_value bool
: The default value of the signal we have been assuming.

value bool
: The value of the signal.

ArchMsg.Warning.bad_reset_in

The signal was not driven at reset by the external system, but was driven after we first started using the default value we had assumed. The external system, if it is going to drive the signal, must do so at first reset so that both sides of the connection can agree on the value of this signal. Fields:

default_value bool
: The default value of the signal we have been assuming.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

value bool
: The value of the signal.

ArchMsg.Warning.contig_bit_gives_too_large_region_for_TxSZ

If the contig bit was used then the size of the contig region would be larger than that indicated by TxSZ. Fields:

desc_kind enum
: Descriptor kind

el enum
: Exception level

input_address unsigned int
: Input address

log2_contig_region_size unsigned int
: Log2(contig region size in bytes)

output_address unsigned int
: Output address

stage_and_level unsigned int
: The stage in bits [7:4] and the level in [3:0].

vmid unsigned int
: VMID if appropriate

ArchMsg.Warning.contig_bit_has_inconsistent_input_and_output_address

If the contig bit was used then the some bits of the output address held in the descriptor and the input address must match. Fields:

desc_kind enum
: Descriptor kind

el enum
: Exception level

input_address unsigned int
: Input address

log2_contig_region_size unsigned int
: Log2(contig region size in bytes)

match_mask unsigned int
: Bits that must match.

output_address unsigned int
: Output address

stage_and_level unsigned int
: The stage in bits [7:4] and the level in [3:0].

vmid unsigned int
: VMID if appropriate

ArchMsg.Warning.fetch_from_memory_type_not_supporting_httu

A descriptor fetch from an HTTU-enabled translation regime to an unsupported memory type was made. Whilst the fetch itself may succeed, if an update to the descriptor was attempted then it would fail. Fields:

address unsigned int
: The address of the descriptor fetch.

desc_inner enum
: Inner cacheability of descriptor

desc_outer enum
: Outer cacheability of descriptor

desc_prot enum
: The instruction/data, bus-ns/s and privileged/user marking

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

desc_sh enum
: Shareability of descriptor

ssd_ns bool
: Is the Security State Determination of the transaction non-secure.

stage unsigned int
: The stage at which we had a problem.

streamid unsigned int
: The StreamID of the transaction

substreamid unsigned int
: The SubstreamID of the transaction (or ~0u if no substreamid)

trans_id unsigned int
: Transaction id.

ArchMsg.Warning.msi_address_truncated

An MSI was generated, but the address was silently truncated due to the limited downstream address bus width. Fields:

pmcg_index_and_counter unsigned int
: If this is a PMCG interrupt, then the top 16 bits are the PMCG index, the lower 16 bits are counter index.

ras_error_group_id_node_id_and_record_index_in_node unsigned int
: If this is a RAS interrupt then this is the error group record id in bits[23:16], the node_id in [15:8] and the record index in the node in [7:0].

truncated_address unsigned int
: The truncated address.

untruncated_address unsigned int
: The untruncated address.

which enum
: Which MSI this is

ArchMsg.Warning.msi_lost

An MSI was attempted to be sent, but the implementation doesn't support it. Fields:

id unsigned int
: ID of this interrupt transaction.

kind enum
: What kind of interrupt.

ArchMsg.Warning.pmcg_non_secure_world

A commentary on potential mis-programming of the PMCG when transitioning the PMCG from the secure world to the non-secure. Fields:

commentary string
: The commentary.

ArchMsg.Warning.pmcg_programming_violates_security

A commentary on potential mis-programming of the PMCG when transitioning the PMCG between non-secure and secure worlds. This is under the assumption that a non-secure asynchronous agent could be writing to the PMCG. Fields:

commentary string
: The commentary.

ArchMsg.Warning.priq_auto_response_failed_to_find_STE

The PRIQ was going to generate an auto-response, but failed to find an STE and so is returning a Failure message to the EndPoint which should disable the PRI interface of the EndPoint. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

trans_id unsigned int
: The ID of the PPR that is being lost.

why enum
: Why the PPR is being lost.

ArchMsg.Warning.priq_overflow_bad_acking

Indicates that an overflow condition was acknowledged by writing to:- SMMU_PRIQ_CONS.OVACKFLG but an overflow condition did not exist. Fields:

new_ovackflg bool
: The new value of the SMMU_PRIQ_CONS.OVACKFLG.

ArchMsg.Warning.priq_smmuen_forces_effective_priqen_low

If SMMUEN == 0, then the effective value of PRIQEN is 0. This warning is triggered when PRIQEN == 1 && SMMUEN == 0; which may not be what was intended. The PRIQ cannot be active if SMMUEN == 0.

ArchMsg.Warning.sev_lost

A SEV was lost because it isn't supported according to SMMU_IDR0.SEV. Fields:

why enum
: Why the SEV was generated.

ArchMsg.Warning.warning

These messages are about unusual (but not necessarily incorrect) activity occurring on the SMMU. Fields:

output string
: The stream output

EVENTQ_config

Configuration of interrupt updated. Fields:

address unsigned int
: Address of the MSI.

data unsigned int
: Data payload of the MSI.

irqen bool
: Appropriate bit of SMMU_s_IRQ_CTRL controlling this interrupt.

memattr enum
: Memory type

ns bool
: Non-secure bus attribute.

ra bool
: read-allocate attribute.

sh enum
: Shareability.

smmu_idr0_msi bool
: Are MSIs supported for this security world.

tr bool
: transient attribute

wa bool
: write-allocate attributes.

EVENTQ_irqen

The IRQEN has been changed. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

new_value bool
The new value of IRQEN.

EVENTQ_irqen_ack

The IRQEN change has been acked. Fields:

new_value bool
The new value of IRQEN.

GERROR_config

Configuration of interrupt updated. Fields:

address unsigned int
Address of the MSI.
data unsigned int
Data payload of the MSI.
irqen bool
Appropriate bit of SMMU_s_IRQ_CTRL controlling this interrupt.
memattr enum
Memory type
ns bool
Non-secure bus attribute.
ra bool
read-allocate attribute.
sh enum
Shareability.
smmu_idr0_msi bool
Are MSIs supported for this security world.
tr bool
transient attribute
wa bool
write-allocate attributes.

GERROR_irqen

The IRQEN has been changed. Fields:

new_value bool
The new value of IRQEN.

GERROR_irqen_ack

The IRQEN change has been acked. Fields:

new_value bool
The new value of IRQEN.

PRIQ_config

Configuration of interrupt updated. Fields:

address unsigned int
Address of the MSI.
data unsigned int
Data payload of the MSI.
irqen bool
Appropriate bit of SMMU_s_IRQ_CTRL controlling this interrupt.
memattr enum
Memory type

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ns bool
: Non-secure bus attribute.

ra bool
: read-allocate attribute.

sh enum
: Shareability.

smmu_idr0_msi bool
: Are MSIs supported for this security world.

tr bool
: transient attribute

wa bool
: write-allocate attributes.

PRIQ_irqen

The IRQEN has been changed. Fields:

new_value bool
: The new value of IRQEN.

PRIQ_irqen_ack

The IRQEN change has been acked. Fields:

new_value bool
: The new value of IRQEN.

SMMU_CR0ACK_SMMUEN_hazarded_by_priq

The SMMU_CR0ACK.SMMUEN cannot acknowledge the change to SMMUEN because there are outstanding PRIQ writes.

SMMU_CR0ACK_SMMUEN_update

The acknowledge to SMMU_CR0.SMMUEN was performed. Fields:

new_value bool
: The new value of the register.

SMMU_CR0_SMMUEN_old_set_complete

A set of transactions associated with the old value of SMMU_CR0.SMMUEN completed. Fields:

last bool
: This is the last set completing.

old_value bool
: The old value of the set completing.

SMMU_CR0_SMMUEN_write

A write to SMMU_CR0.SMMUEN occurred. Fields:

new_value bool
: The new value of the register.

old_value bool
: The old value of the register.

SMMU_GBPA_old_set_complete

A set of transactions associated with the old value of SMMU_GBPA completed. Fields:

last bool
: This is the last set completing.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

old_value unsigned int
The old value of the set completing.

SMMU_GBPA_update

The Update flag to SMMU_GBPA was lowered. Fields:

new_value unsigned int
The new value of the register.

SMMU_GBPA_write

A write to SMMU_GBPA occurred. Fields:

new_value unsigned int
The new value of the register.
old_value unsigned int
The old value of the register.

SMMU_S_CR0ACK_SMMUEN_update

The acknowledge to SMMU_S_CR0.SMMUEN was performed. Fields:

new_value bool
The new value of the register.

SMMU_S_CR0_SMMUEN_old_set_complete

A set of transactions associated with the old value of SMMU_S_CR0.SMMUEN completed. Fields:

last bool
This is the last set completing.
old_value bool
The old value of the set completing.

SMMU_S_CR0_SMMUEN_write

A write to SMMU_S_CR0.SMMUEN occurred. Fields:

new_value bool
The new value of the register.
old_value bool
The old value of the register.

SMMU_S_GBPA_old_set_complete

A set of transactions associated with the old value of SMMU_S_GBPA completed. Fields:

last bool
This is the last set completing.
old_value unsigned int
The old value of the set completing.

SMMU_S_GBPA_update

The Update flag to SMMU_S_GBPA was lowered. Fields:

new_value unsigned int
The new value of the register.

SMMU_S_GBPA_write

A write to SMMU_S_GBPA occurred. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

new_value unsigned int
: The new value of the register.

old_value unsigned int
: The old value of the register.

S_EVENTQ_config

Configuration of interrupt updated. Fields:

address unsigned int
: Address of the MSI.

data unsigned int
: Data payload of the MSI.

irqen bool
: Appropriate bit of SMMU_s_IRQ_CTRL controlling this interrupt.

memattr enum
: Memory type

ns bool
: Non-secure bus attribute.

ra bool
: read-allocate attribute.

sh enum
: Shareability.

smmu_idr0_msi bool
: Are MSIs supported for this security world.

tr bool
: transient attribute

wa bool
: write-allocate attributes.

S_EVENTQ_irqen

The IRQEN has been changed. Fields:

new_value bool
: The new value of IRQEN.

S_EVENTQ_irqen_ack

The IRQEN change has been acked. Fields:

new_value bool
: The new value of IRQEN.

S_GERROR_config

Configuration of interrupt updated. Fields:

address unsigned int
: Address of the MSI.

data unsigned int
: Data payload of the MSI.

irqen bool
: Appropriate bit of SMMU_s_IRQ_CTRL controlling this interrupt.

memattr enum
: Memory type

ns bool
: Non-secure bus attribute.

ra bool
: read-allocate attribute.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

sh enum
: Shareability.

smmu_idr0_msi bool
: Are MSIs supported for this security world.

tr bool
: transient attribute

wa bool
: write-allocate attributes.

S_GERROR_irqen

The IRQEN has been changed. Fields:

new_value bool
: The new value of IRQEN.

S_GERROR_irqen_ack

The IRQEN change has been acked. Fields:

new_value bool
: The new value of IRQEN.

TLB.tlb_commentary

A commentary from the cache about what is going on. Fields:

output string
: The stream output.

TLB.tlb_entry_invalidated

An entry in the cache has been invalidated. Fields:

entry_id unsigned int
: The entry id that is being invalidated.

reason enum
: Why the entry is being invalidated.

atc_inv_nop

The CMD_ATC_INV command is ignored as a NOP. This may be emitted multiple times if the CMD_ATC_INV is being ignored for multiple reasons. Fields:

address unsigned int
: Untranslated Address to invalidate.

cmd_id unsigned int
: Command id, if top-bit is set then was issued from the Non-secure CMDQ.

cons unsigned int
: CONS of the command.

global bool
: Global flag.

log2_size_in_bytes unsigned int
: Log 2 of the size in bytes of the region to invalidate.

streamid unsigned int
: StreamID in the invalidate command.

substreamid unsigned int
: SubstreamID or ~0u if SSValid == false.

why enum
: Why was NOPped

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

atos_complete_fault

The ATOS operation completed with a fault. Fields:

effective_st1translate bool
: Because of the settings and/or the ATOS type then the effective st1translate can be different.

fault_faddr unsigned int
: The fault FADDR.

fault_faultcode enum
: The fault code.

fault_reason enum
: The fault reason.

ssd_ns bool
: This is a non-secure ATOS operation.

st1translate bool
: The StreamID/SubstreamID combination should have been translated by stage 1.

st2translate bool
: The translation suffered a stage 2 translation.

state enum
: The final transaction state of the ATOS operation.

vatos_vmid unsigned int
: The VMID of the VATOS operation, or ~0u if not a VATOS operation.

atos_complete_fault_inv_req

The ATOS operation completed, faulted and generated an INV_REQ response. Fields:

ssd_ns bool
: This is a non-secure ATOS operation.

vatos_vmid unsigned int
: The VMID of the VATOS operation, or ~0u if not a VATOS operation.

why enum
: Why the ATOS operation generated an INV_REQ

atos_complete_success

The ATOS operation completed successfully. Fields:

base_addr unsigned int
: The actual base address of region.

effective_st1translate bool
: Because of the settings and/or the ATOS type then the effective st1translate can be different.

par_addr unsigned int
: The PAR.ADDR field.

par_mair unsigned int
: The memory attributes encoded as a MAIR

par_ns bool
: The PAR.NS field, for an SSD-ns request then this will always be 0.

par_sh enum
: Shareability

par_size bool
: The PAR.Size field.

size_in_bytes unsigned int
: The actual size in bytes of the region.

ssd_ns bool
: This is a non-secure ATOS operation.

st1translate bool
: The StreamID/SubstreamID combination should have been translated by stage 1.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

st2translate bool
: The translation suffered a stage 2 translation.

state enum
: The final transaction state of the ATOS operation.

vatos_vmid unsigned int
: The VMID of the VATOS operation, or ~0u if not a VATOS operation.

atos_run_set

The SMMU_s_GATOS_CTRL.RUN field was set to start the ATOS operation. Fields:

ssd_ns bool
: This is a non-secure ATOS operation.

vatos bool
: This is a VATOS operation

atos_starting

The ATOS operation is starting. Fields:

addr unsigned int
: The input address to the ATOS operation.

httui bool
: Inhibit HTTU update.

ind bool
: Instruction Data.

pnu bool
: Privileged not User.

rnw bool
: Read not Write.

ssd_ns bool
: This is a non-secure ATOS operation.

ssec bool
: If this is a secure ATOS operation then this is if it is secure or not.

streamid unsigned int
: The StreamID requested.

substreamid unsigned int
: The SubstreamID requested, or ~0u if no SubstreamID.

type enum
: The requested ATOS type.

vatos_vmid unsigned int
: The VMID of the VATOS operation, or ~0u if not a VATOS operation.

cd_cc.CD_commentary

A commentary from the cache about what is going on. Fields:

output string
: The stream output.

cd_cc.CD_entry_invalidated

An entry in the cache has been invalidated. Fields:

entry_id unsigned int
: The entry id that is being invalidated.

reason enum
: Why the entry is being invalidated.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

cd_entry_allocated

An CD entry has been allocated. Fields:

cd string
: A textual description of the CD.

entry_id unsigned int
: The id of the cache entry. Top bit is set if is a debug entry.

ns enum
: For the non-secure world

streamid unsigned int
: The StreamID of the entry this will match.

substreamid unsigned int
: The SubstreamID of the entry this will match. This may be zero for transactions without a SubstreamID.

conf_reset_of_SMMU_GBPA_ABORT

The reset value of SMMU_GBPA.ABORT. Fields:

value bool
: The value of the signal.

conf_reset_of_SMMU_S_GBPA_ABORT

The reset value of SMMU_S_GBPA.ABORT. Fields:

value bool
: The value of the signal.

conf_system_supports_btm

The system supports BTM. Fields:

value bool
: The value of the signal.

conf_system_supports_cohacc

The system supports cohacc. Fields:

value bool
: The value of the signal.

conf_system_supports_httu

The system supports HTTU. Fields:

value bool
: The value of the signal.

conf_system_supports_sev

The system supports SEV. Fields:

value bool
: The value of the signal.

dvm_tlbinvalidate_complete

The DVM TLB Invalidate message completed. Fields:

id unsigned int
: The unique id of this DVM message.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ok bool
: The DVM message was OK.

dvm_tlbinvalidate_received

A DVM message for a TLB Invalidate has been received. Fields:

address unsigned int
: The VA or IPA to use if match_address.

asid unsigned int
: The ASID to match if match_asid.

by_ipa bool
: The operation is for an IPA operation if match_address.

id unsigned int
: The unique id of this DVM message.

ignored enum
: The DVM message was ignored

last_level bool
: The operation is for last level if supported.

match_address bool
: Match the address field.

match_asid bool
: Match the asid field.

match_vmid bool
: Match the vmid field.

num unsigned int
: If a range operation, the NUM field. If a single-address operation this is 0.

prot enum
: The protection level for which this TLB Invalidate will operate on.

security_world enum
: The security world that this will apply to.

smmu_scale unsigned int
: If a range operation, then the SCALE field with the meaning in the SMMU architecture which is different to the PE architecture. If a single-address operation this is 0.

stage1_only bool
: The operation is for stage 1 only if supported.

tg enum
: If a single-address or address-range operation, then the Translation Granule hint. Address-range operations always supply a Translation Granule.

translation_table_level enum
: The leaf level of the translation table.

vmid unsigned int
: The VMID to match if match_vmid.

found_tlb_entry_has_different_aset

Architecturally, a particular ASID either should be ASET0 or ASET1. However, we have managed to find a TLB entry that has a different ASET than that which we were searching for. This indicates a programming error. You should examine all contexts with this particular ASID/VMID and ensure they are consistent. Fields:

aset unsigned int
: The ASET of this entry.

asid unsigned int
: ASID if appropriate.

index unsigned int
: Index of TLB entry.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

input_start_address unsigned int
: The start address of the input range that this matches.

scheme enum
: The tagging scheme used

ssd_ns enum
: Security State

vmid unsigned int
: VMID if appropriate.

gerror_inverted

A GERROR bit was attempted to be inverted to record a fault. Fields:

already_different_to_gerrorn bool
: True if the bit was already different to the corresponding bit in GERRORN and so it was not inverted.

gerrorn unsigned int
: GERRORN register value.

interrupt_potentially_sent bool
: An interrupt is potentially sent, depending on IRQEN.
GERROR(N).MSI_GERROR_ABT_ERR does not send an interupt.

new_gerror unsigned int
: GERROR register value after any invertion.

which enum
: Which bit was attempted to be inverted.

gerrorn_acknowledge

A GERROR was acknowledged by SW. Fields:

which enum
: Which GERROR was acknowledged.

httu_update_abandoned_update

The HTTU update of a descriptor in memory was potentially possible, but it was behind an update that failed to apply cleanly. Fields:

AF enum
: Whether an AF update should/could occur.

DBM enum
: Whether a DBM update should/could occur.

address unsigned int
: Address of HTTU update.

httu_inner enum
: Inner cacheability of descriptor to update

httu_outer enum
: Outer cacheability of descriptor to update

httu_prot enum
: The instruction/data, bus-ns/s and privileged/user marking

httu_sh enum
: Shareability of descriptor to update

mpam_pmg_and_partid unsigned int
: Bits[23:16] are the MPAM PMG. Bits[15:0] are the physical PARTID.

original_descriptor unsigned int
: The original descriptor that the model observed.

stage_and_level unsigned int
: The stage (bits[7:4]) and level (bits[3:0]).

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

trans_id unsigned int
: The transaction id that this HTTU update is associated with.

what enum
: What this descriptor represents

httu_update_end_update

The attempted update of a descriptor in memory has occurred. Fields:

is_big_endian bool
: The descriptor is big-endian in memory.

original_descriptor unsigned int
: The original descriptor value.

result enum
: The result of the attempt to update.

trans_id unsigned int
: The transaction id that this HTTU update is associated with.

try_to_change_to_descriptor unsigned int
: The new descriptor value that tried to replace the original.

value_that_was_in_memory unsigned int
: The value that the compare-and-swap operation returned as the value that was in memory.

httu_update_not_done

A discretionary HTTU update could occur and the implementation choose not to do it. A mandatory_do_if_linked_performed means that this is a stage 2 leaf descriptor that maps a subsequent stage 1 leaf descriptor whose update is discretionary and if that subsequent discretionary update is going to occur then this update becomes mandatory. Fields:

AF enum
: Whether an AF update should/could occur.

DBM enum
: Whether a DBM update should/could occur.

address unsigned int
: Address of HTTU update.

httu_inner enum
: Inner cacheability of descriptor to update

httu_outer enum
: Outer cacheability of descriptor to update

httu_prot enum
: The instruction/data, bus-ns/s and privileged/user marking

httu_sh enum
: Shareability of descriptor to update

mpam_pmg_and_partid unsigned int
: Bits[23:16] are the MPAM PMG. Bits[15:0] are the physical PARTID.

original_descriptor unsigned int
: The original descriptor that the model observed.

stage_and_level unsigned int
: The stage (bits[7:4]) and level (bits[3:0]).

trans_id unsigned int
: The transaction id that this HTTU update is associated with.

what enum
: What this descriptor represents

httu_update_start_update

An HTTU update could occur and the implementation chose to try it. A mandatory_do_if_linked_performed means that this is a stage 2 leaf descriptor that maps a subsequent

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

stage 1 leaf descriptor whose update is discretionary and if that subsequent discretionary update is going to occur then this update becomes mandatory. Fields:

AF enum
: Whether an AF update should/could occur.

DBM enum
: Whether a DBM update should/could occur.

address unsigned int
: Address of HTTU update.

httu_inner enum
: Inner cacheability of descriptor to update

httu_outer enum
: Outer cacheability of descriptor to update

httu_prot enum
: The instruction/data, bus-ns/s and privileged/user marking

httu_sh enum
: Shareability of descriptor to update

is_big_endian bool
: The descriptor will be written to memory as big-endian.

mpam_pmg_and_partid unsigned int
: Bits[23:16] are the MPAM PMG. Bits[15:0] are the physical PARTID.

original_descriptor unsigned int
: The original descriptor that the model observed.

stage_and_level unsigned int
: The stage (bits[7:4]) and level (bits[3:0]).

trans_id unsigned int
: The transaction id that this HTTU update is associated with.

try_to_change_to_descriptor unsigned int
: The new descriptor value that is going to try replace the original.

what enum
: What this descriptor represents

will_do_AF bool
: What the implementation chose to do for the AF update.

will_do_DBM bool
: What the implementation chose to do for the DBM update.

interrupt_returned

An interrupt/MSI returned from downstream. Fields:

InD_NS_PnU enum
: Instruction/Data, Non-secure/Secure, Privileged/User.

address unsigned int
: Address of the MSI.

data unsigned int
: Data payload of the MSI.

id unsigned int
: ID of this interrupt transaction.

inner_cache enum
: The actual attributes of the access.

kind enum
: What kind of interrupt.

mpam_ns bool
: The security state associated with the MPAM PARTID/PMG.

mpam_pmg_and_partid unsigned int
: Bits[23:16] are the MPAM PMG. Bits[15:0] are the physical PARTID.

msi_valid bool
: MSI will attempt to be sent.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ok bool
: Did the access return OK or an abort?

outer_cache enum
: The actual attributes of the access.

pmcg_index_and_counter unsigned int
: If this is a PMCG interrupt, then the top 16 bits are the PMCG index, the lower 16 bits are counter index.

ras_error_group_id_node_id_and_record_index_in_node unsigned int
: If this is a RAS interrupt then this is the error group record id in bits[23:16], the node_id in [15:8] and the record index in the node in [7:0].

sh enum
: Shareability

interrupt_sent

An interrupt is raised. If it sends an MSI then this is _after_ any device-dependent transform on the architectural attributes and so may differ from what is programmed. Fields:

InD_NS_PnU enum
: Instruction/Data, Non-secure/Secure, Privileged/User.

address unsigned int
: Address of the MSI.

data unsigned int
: Data payload of the MSI.

id unsigned int
: ID of this interrupt transaction.

inner_cache enum
: The actual attributes of the access.

kind enum
: What kind of interrupt.

mpam_ns bool
: The NS state of the MPAM PARTID and MPAM PMG.

mpam_pmg_and_partid unsigned int
: Bits[23:16] are the MPAM PMG. Bits[15:0] are the physical PARTID.

msi_valid bool
: MSI will attempt to be sent.

outer_cache enum
: The actual attributes of the access.

pmcg_index_and_counter unsigned int
: If this is a PMCG interrupt, then the top 16 bits are the PMCG index, the lower 16 bits are counter index.

ras_error_group_id_node_id_and_record_index_in_node unsigned int
: If this is a RAS interrupt then this is the error group record id in bits[23:16], the node_id in [15:8] and the record index in the node in [7:0].

sh enum
: Shareability

l1cd_cc.L1CD_commentary

A commentary from the cache about what is going on. Fields:

output string
: The stream output.

l1cd_cc.L1CD_entry_invalidated

An entry in the cache has been invalidated. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

entry_id unsigned int
: The entry id that is being invalidated.

reason enum
: Why the entry is being invalidated.

l1cd_entry_allocated

An L1 CD entry has been allocated. Fields:

entry_id unsigned int
: The id of the cache entry. Top bit is set if is a debug entry.

normalised_substreamid unsigned int
: The first SubstreamID of the range of SubstreamIDs that this L1CD entry will match.

ns enum
: For the non-secure world

pa_l2 unsigned int
: The PA of the L2 CD table. This is L2Ptr << 12.

streamid unsigned int
: The StreamID this CD is for.

valid bool
: Is the entry valid.

l1ste_cc.L1STE_commentary

A commentary from the cache about what is going on. Fields:

output string
: The stream output.

l1ste_cc.L1STE_entry_invalidated

An entry in the cache has been invalidated. Fields:

entry_id unsigned int
: The entry id that is being invalidated.

reason enum
: Why the entry is being invalidated.

l1ste_entry_allocated

An L1 STE entry has been allocated. Fields:

entry_id unsigned int
: The id of the cache entry. Top bit is set if is a debug entry.

first_streamid_of_range unsigned int
: The first StreamID of the range of StreamIDs that this L1STE entry will match.

ns enum
: For the non-secure world

num_entries_in_l2 unsigned int
: The number of entries in the L2 table. This is 2**(Span-1) or 0 if invalid.

pa_l2 unsigned int
: The PA of the L2 ST table. This is L2Ptr << 6 and aligned to the size of the table.

ns_cmd_sync_completed_irq

Non-secure CMD_SYNC completed its IRQ action. Fields:

cmd_id unsigned int
: The ID of the command being issued.

cons_incl_wrap unsigned int
: CONS pointer of CMD_SYNC.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ns_cmd_sync_completed_no_action

Non-secure CMD_SYNC completed, there was no CMD_SYNC action (SIG_NONE). Fields:

cmd_id unsigned int
: The ID of the command being issued.

cons_incl_wrap unsigned int
: CONS pointer of CMD_SYNC.

ns_cmd_sync_completed_sev

Non-secure CMD_SYNC completed its action of SIG_SEV. Fields:

cmd_id unsigned int
: The ID of the command being issued.

cons_incl_wrap unsigned int
: CONS pointer of CMD_SYNC.

ns_cmd_sync_completed_with_error

Non-secure CMD_SYNC completed, however, there was an error associated with the completion of the CMD_SYNC. Fields:

cerror enum
: The error associated with the CMD_SYNC

cmd_id unsigned int
: The ID of the command being issued.

cons_incl_wrap unsigned int
: CONS pointer of CMD_SYNC.

ns_cmd_sync_issued

Non-secure CMD_SYNC has now completed issuing and we now know if something is hazarding the CMD_SYNC from completing. Fields:

cmd_id unsigned int
: The ID of the command being issued.

cons_incl_wrap unsigned int
: CONS pointer of CMD_SYNC.

hazarded bool
: The CMD_SYNC is currently hazarded from completing.

ns_cmd_sync_issuing

Non-secure CMD_SYNC is starting to issue, we are going to start accounting for which things have to be part of the prior set of things that have to complete before the CMD_SYNC can complete. Fields:

cmd_id unsigned int
: The ID of the command being issued.

cons_incl_wrap unsigned int
: CONS pointer of CMD_SYNC.

ns_cmd_sync_starting_completion_action_irq

Non-secure CMD_SYNC starting IRQ completion action. Fields:

cmd_id unsigned int
: The ID of the command being issued.

cons_incl_wrap unsigned int
: CONS pointer of CMD_SYNC.

msiaddress unsigned int
: The MSI address to use (or 0 if not sending an MSI).

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

msidata unsigned int
: The MSI data to send.

ns_cmd_sync_starting_completion_action_sev

Non-secure CMD_SYNC starting its completion action of SIG_SEV. Fields:

cmd_id unsigned int
: The ID of the command being issued.

cons_incl_wrap unsigned int
: CONS pointer of CMD_SYNC.

ns_cmdq_cmd_sync_error

The CMD_SYNC can fault if an ATC Invalidate timeout occurs. The timeout will be attached to the next CMD_SYNC on the queue on which the invalidate was launched. Fields:

cerror enum
: The CERROR to attach.

cmd_queue unsigned int
: The top bits indicate the CMDQ id. These bits are attached to all cmd_ids generated for this queue.

ns_cmdq_cmd_sync_error_delivered

The CMD_SYNC can fault if an ATC Invalidate timeout occurs. This is the CMD_SYNC delivering that fault. Fields:

cerror enum
: The CERROR to attach.

cmd_id unsigned int
: Command ID of the CMD_SYNC

ns_cmdq_ecmdq_enack

Trace the changes to ENACK. Fields:

cerror enum
: The error being exposed, if any.

cmd_queue unsigned int
: The top bits indicate the CMDQ id. These bits are attached to all cmd_ids generated for this queue.

enack bool
: The value of ENACK.

forced_error bool
: The queue was enabled with ERR != ERRACK and so the implementation chooses to trigger the error reporting mechanism.

ns_cmdq_issue

A command is actually being executed. Fields:

cmd_id unsigned int
: Command ID of the command being executed.

cons unsigned int
: CONS of the command.

what string
: What is being executed.

ns_cmdq_pointers_state

The command queue pointers. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

cmd_queue unsigned int
: The top bits indicate the CMDQ id. These bits are attached to all cmd_ids generated for this queue.

disable_fetch bool
: The model has decided that as you programmed PROD/CONS inconsistently that it would not fetch as it has no confidence as to which are commands and which are not.

q_issuable_and_issued_cons unsigned int
: The CONS pointer of the region that is issuable and/or issued (including wrap).

q_issuable_and_issued_prod unsigned int
: The PROD pointer of the region that is issuable and/or issued (including wrap).

q_programmer_cons unsigned int
: The CONS pointer as seen by the programmer (including wrap).

q_programmer_prod unsigned int
: The PROD pointer as seen by the programmer (including wrap).

q_unfetched_cons unsigned int
: The CONS pointer of the region unfetched (including wrap).

q_unfetched_prod unsigned int
: The PROD pointer of the region unfetched (including wrap).

ns_cmdq_state

A description of what the CMDQ can do now. Fields:

can_cerror bool
: There is a CERROR that can be recognised.

can_cmd_sync bool
: There is a CMD_SYNC that can be recognised.

can_fetch bool
: The SMMU can fetch

can_issue bool
: There are fetched commands that are eligible to be issued.

can_update_cons bool
: The programmer view of CMDQ_CONS can be moved to say some have been consumed.

cmd_queue unsigned int
: The top bits indicate the CMDQ id. These bits are attached to all cmd_ids generated for this queue.

cmd_sync enum
: Why a CMD_SYNC cannot be issued

cmdqen bool
: CMDQEN, when enabled then the CMDQ can fetch commands

cmdqen_1_to_0_unacknowledged bool
: CMDQEN has been set to 0, but we have yet to acknowledge it.

current-real-cerror enum
: The current CERROR that would be visible to the programmer if there was an error.

fetch-disabled_due_to_programmer_error bool
: The PROD/CONS pointer was inconsistent and the model decided to stop processing commands.

fetch-number_of_unfetched_commands unsigned int
: The number of commands that have yet to be fetched from memory, or ~0u if CMDQEN == 0.

fetch-state enum
: The state of the fetch state machine

fetch-too_many_commands_outstanding_so_not_fetching_any_more bool
: If there are too may issued commands, or pending to be issued then we won't fetch any more.

issue-number_of_issuable_commands unsigned int
: The number of commands that have been fetched but not yet issued.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

issue-number_of_unissuable_commands unsigned int
: This might include commands behind an illegal command, or behind a CMD_SYNC. Under some circumstances this can include a command representing an external abort.

issue-state enum
: The internal issued state machine state

outstanding_fetches_since_reset unsigned int
: The number of outstanding fetches.

ns_eventq_adding_to_cmd_sync_prior_set

The event created by the specified transaction id must become visible to the programmer before the CMD_SYNC can complete. Fields:

trans_id unsigned int
: The transaction id that caused the event.

ns_eventq_auto_retry_stalled_transaction

A transaction that was stalled but not yet reported on the Event Queue was retried. The retry can happen if the transaction didn't write because it was de-duplicated behind a reported stalling transaction and that stalling transaction was CMD_RESUMEd, or if the queue was unwriteable and the queue is now writeable. Fields:

trans_id unsigned int
: Transaction id of the auto-retried transaction.

why enum
: Why the transaction retried.

ns_eventq_cmd_sync_unhazarded

The CMD_SYNC has been unhazarded as the prior set is empty and as much of it as required is visible in the programmers' view. Fields:

prod_incl_wrap unsigned int
: The prod index including wrap bit that was required to become visible.

prog_prod_incl_wrap unsigned int
: The current programmer-visible prod including wrap bit.

ns_eventq_eventqen_and_eventqenack

Trace the values of EVENTQEN and EVENTQENACK as they change. Fields:

about enum
: Some more information about the queue state.

cons_incl_wrap unsigned int
: The CONS pointer (including wrap bits).

eventqen bool
: EVENTQEN.

eventqenack bool
: EVENTQENACK.

prod_incl_wrap unsigned int
: The PROD pointer (including wrap bits).

ns_eventq_external_abort

A particular eventq record got an external abort. The event will subsequently appear in the ns/s_eventq_lost_event_records trace. Fields:

prod unsigned int
: The prod pointer (excluding wrap bit) of the event record that aborted.

trans_id unsigned int
: The transaction id that produced the event record that aborted.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ns_eventq_lost_event_records

Event records that are lost and never record appear on this trace. Fields:

dword0 unsigned int
: Double word 0 of event.

dword1 unsigned int
: Double word 1 of event.

dword2 unsigned int
: Double word 2 of event.

dword3 unsigned int
: Double word 3 of event.

event enum
: Event code that was thrown away.

prod unsigned int
: The PROD index of the event if appropriate or ~0u otherwise.

reason enum
: The reason the record was lost.

trans_id unsigned int
: The transaction id that this corresponds to.

ns_eventq_overflow_acknowledged

When events are lost then they appear on the eventq_lost_event_records_* trace. This will activate the overflow flag in the corresponding SMMU_s_EVENTQ_PROD register. Software can then acknowledge that flag using the overflow acknowledge flag in SMMU_s_EVENTQ_CONS. Fields:

bad bool
: If true then the programmer has attempted to acknowledge an overflow condition that didn't exist.

count_of_events_lost_due_to_overflow unsigned int
: This is the number of events that were lost because the event queue was full. This does not include any lost because of an EVENTQ_ABT_ERR.

ns_eventq_pending_event_records

Event records that are pending to be dealt with by the eventq state machine. A specific event can be traced multiple times if it becomes part of the prior set of transactions of a CMD_SYNC that has to complete before the CMD_SYNC is allowed to be completed. Fields:

dword0 unsigned int
: Double word 0 of event.

dword1 unsigned int
: Double word 1 of event.

dword2 unsigned int
: Double word 2 of event.

dword3 unsigned int
: Double word 3 of event.

event enum
: Event code.

is_prior_set bool
: The transaction forms part of the prior set of a CMD_SYNC that have to become visible (or thrown away if necessary) before the CMD_SYNC can complete

trans_id unsigned int
: The transaction id that this corresponds to.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ns_eventq_prod_cons_updated

This records when the programmer visible PROD/CONS pointer is updated. More can be visible in memory than are advertised in the programmer-visible PROD pointer. This also traces the latest PROD that could have been advertised. Fields:

mem_prod unsigned int
: The PROD pointer value for all records that are visible in the memory.

num_events_available unsigned int
: The number of events available to the programmer.

prog_cons unsigned int
: The programmer-visible CONS pointer.

prog_prod unsigned int
: The programmer-visible PROD pointer.

what enum
: What changed?

ns_eventq_raising_eventq_abt_err

EVENTQ_ABT_ERR can now be raised.

ns_eventq_removing_from_cmd_sync_prior_set

The event captured by a CMD_SYNC and created by the specified transaction id is now in memory. Fields:

prod unsigned int
: The prod index (no wrap bit) that must become visible to the programmer to see this record. If not head_of_line then the record will not become visible and this field is meaningless.

reason enum
: The reason the event was removed.

trans_id unsigned int
: The transaction id that caused the event.

ns_eventq_stashing_unreported_stalled_transaction_for_auto_retry

A stall event record would have been lost, instead it is stashed away to auto-retry when it can. Fields:

dword0 unsigned int
: Double word 0 of event.

dword1 unsigned int
: Double word 1 of event.

dword2 unsigned int
: Double word 2 of event.

dword3 unsigned int
: Double word 3 of event.

event enum
: Event code that was thrown away.

reason enum
: The reason the record was lost.

trans_id unsigned int
: The transaction id that this corresponds to.

ns_eventq_write_event

An event is committed to being written. Fields:

dword0 unsigned int
: Double word 0 of event.

dword1 unsigned int
: Double word 1 of event.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

dword2 unsigned int
: Double word 2 of event.

dword3 unsigned int
: Double word 3 of event.

event enum
: Event code that was thrown away.

prod unsigned int
: The PROD pointer (without wrap) that this event will be written to.

stall_event bool
: Describes an event for a stalled transaction.

trans_id unsigned int
: Transaction ID that generated this event.

ns_eventq_write_event_visible_in_memory

An event is now visible in memory. Fields:

abort bool
: The write aborted.

dword0 unsigned int
: Double word 0 of event.

dword1 unsigned int
: Double word 1 of event.

dword2 unsigned int
: Double word 2 of event.

dword3 unsigned int
: Double word 3 of event.

event enum
: Event code that was thrown away.

prod unsigned int
: The PROD pointer (without wrap) that this event will be written to.

stall_event bool
: Describes an event for a stalled transaction.

trans_id unsigned int
: Transaction ID that generated this event.

pmcg_irq_config

The interrupt configuration of the Performance Monitor Counter Group (PMCG) changed. Fields:

address unsigned int
: Address of the MSI.

data unsigned int
: Data payload of the MSI.

irqen bool
: SMMU_PMCG_CTRL.IRQEN.

memattr enum
: Memory type

mpam_ns bool
: The NS state of the MPAM PARTID and MPAM PMG

mpam_pmg_and_partid unsigned int
: Bits[23:16] are the MPAM PMG. Bits[15:0] are the physical PARTID.

msi_supported bool
: Are MSIs supported for this security world.

ns bool
: Non-secure bus attribute.

number_of_interrupts_in_flight unsigned int
: The number of interrupts that have been committed to be produced or in flight.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

pmcg_index unsigned int
: Index of the PMCG

sh enum
: Shareability.

smmu_pmcg_gmpam_Update bool
: The SMMU_PMCG_GMPAM.Update flag. Only when this is zero are writes predictable.

ssd_ns bool
: True if this PMCG controlled by the non-secure world.

pmcg_irqen

A trace of SMMU_PMCG_IRQ_CTRL.IRQEN. Fields:

new_value bool
: The new value of IRQEN.

pmcg_index unsigned int
: Index of the PMCG.

ssd_ns bool
: True if this PMCG controlled by the non-secure world.

pmcg_irqen_ack

A trace of SMMU_PMCG_IRQ_CTRLACK.IRQEN. Fields:

new_value bool
: The new value of IRQEN.

pmcg_index unsigned int
: Index of the PMCG.

ssd_ns bool
: True if this PMCG controlled by the non-secure world.

pmcg_merging_interrupts

An interrupt was wanted to be generated, but one was already pending so the two were merged together. Fields:

pmcg_index unsigned int
: Index of the PMCG.

ssd_ns bool
: True if this PMCG controlled by the non-secure world.

pmcg_trigger

A PMCG counter has been triggered. Fields:

counter_index unsigned int
: The index of the counter within the PMCG.

event_id unsigned int
: The event id that has been triggered.

ns_event bool
: The security state associated with the event.

pmcg_index unsigned int
: Index of the PMCG.

streamid unsigned int
: The StreamID associated with the event, if there is one.

pmu_active_counter

Traces what active counters are in a PMCG and what streamids it might filter on. Those counters that trace both secure and non-secure StreamIDs or those that are not filtered by StreamID will appear twice, once for each security world. All active counters for a PMCG are traced one after another. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

begin_streamid unsigned int
: The start StreamID to filter on.

counter_index unsigned int
: The counter index within the PMCG.

end_incl_streamid unsigned int
: The end inclusive StreamID to filter on.

evcnt unsigned int
: The current count.

event_id unsigned int
: The event ID to filter.

ns bool
: Are the StreamIDs non-secure?

pmcg_index unsigned int
: The index of the PMCG that this counter belongs to.

pmu_all_counters_in_pmcg_became_inactive

The PMCG was tracing some events and now is not tracing any. Fields:

pmcg_index unsigned int
: The index of the PMCG.

pmu_capture

For some reason, a capture event occurred. Fields:

pmcg_index unsigned int
: The index of the PMCG that the capture occurred on.

why enum
: Why did the capture occur?

pmu_counter_configured_to_use_unsupported_event

An enabled counter was configured to use a unsupported event. Fields:

counter_index unsigned int
: The counter index within the PMCG.

event_id unsigned int
: The unsupported event id.

pmcg_index unsigned int
: The index of the PMCG that this counter belongs to.

pmu_counter_overflowed

A counter in a particular PMCG overflowed. Fields:

already_overflowed bool
: True if the overflow flag was already set.

capture bool
: True if it captured the other counter values.

counter_index unsigned int
: The counter index within the PMCG.

interrupt bool
: True if going to attempt to generate an interrupt.

interrupt_action enum
: The interrupt action that is going to occur

pmcg_index unsigned int
: The index of the PMCG that this counter belongs to.

ssd_ns bool
: The PMCG is controlled by the Non-secure world.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

priq_auto_response

A PRIQ auto response is generated. Fields:

prgindex unsigned int
: The PRG Index of the response.

resp enum
: The response to send.

streamid unsigned int
: The StreamID of the response.

substreamid unsigned int
: The PASID/SubstreamID attached to the response, or ~0u if there is none.

trans_id unsigned int
: The ID of the PPR that is being lost.

why enum
: Why the PPR is being lost.

priq_lost_ppr

The PRIQ is not reporting a PPR into the queue. It may or may not generate an auto-response. Fields:

is_stop_marker bool
: The PPR being lost is a Stop Marker.

resp enum
: What, if any, auto-response will be.

trans_id unsigned int
: The ID of the PPR that is being lost.

why enum
: Why the PPR is being lost

priq_overflow_acking

Indicates that an overflow condition was acknowledged by writing to:- SMMU_PRIQ_CONS.OVACKFLG. Fields:

new_ovackflg bool
: The new value of the SMMU_PRIQ_CONS.OVACKFLG.

priq_overflow_asserting

Indicates that we are toggling the SMMU_PRIQ_PROD.OVFLG because we lost a PRI request due to the PRIQ being full and an existing overflow condition does not already exist. Fields:

new_ovflg bool
: The new value of the SMMU_PRIQ_PROD.OVFLG.

trans_id unsigned int
: The transaction ID of the PPR that caused the overflow.

priq_received

A PRIQ Request has been received and is queued waiting processing. This does not mean that it has been written to the PRIQ but has been placed in a pending queue awaiting a decision about what to do about it. Fields:

L bool
: The Last in the Page Request Group (PRG).

R bool
: Read permission is requested.

W bool
: Read permission is requested.

prgindex unsigned int
: The Page Request Group (PRG) index that is being used to identify this request to the EndPoint.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ssd_ns bool
: The SSD of the StreamID. PRIQ should only receive _non-secure_ PRI Requests.

stop_marker bool
: If LRW == 0b100 then this indicates this is a stop marker.

streamid unsigned int
: The StreamID this request is associated with.

substreamid unsigned int
: The SubstreamID (PASID) this request is associated with, of ~0u if no SubstreamID.

substreamid_P bool
: If has a SubstreamID (!= ~0u) then Privileged Mode Requested.

substreamid_X bool
: If has a SubstreamID (!= ~0u) then Execution Requested (R must also be 1).

trans_id unsigned int
: The transaction ID that identifies this request.

untranslated_address unsigned int
: The Untranslated Address that this is requesting.

priq_write_aborted

A PRIQ write aborted. The PRIQ now goes into an error state and will start auto-responding to PRI requests. Fields:

prod_incl_wrap unsigned int
: The PROD including the Wrap bit where we were trying to write.

trans_id unsigned int
: The transaction ID of the PPR that aborted.

priq_write_ok

A PRIQ write completed OK. Fields:

prod_incl_wrap unsigned int
: The PROD including the Wrap bit where we wrote..

trans_id unsigned int
: The transaction ID of the PPR.

priq_write_start

A PRIQ request has been received and is going to be attempt to be written to the queue. Fields:

L bool
: The Last in the Page Request Group (PRG).

R bool
: Read permission is requested.

W bool
: Read permission is requested.

prgindex unsigned int
: The Page Request Group (PRG) index that is being used to identify this request to the EndPoint.

prod_incl_wrap unsigned int
: PROD position including the wrap bit.

stop_marker bool
: If LRW == 0b100 then this indicates this is a stop marker.

streamid unsigned int
: The StreamID this request is associated with.

substreamid unsigned int
: The SubstreamID (PASID) this request is associated with, of ~0u if no SubstreamID.

substreamid_P bool
: If has a SubstreamID (!= ~0u) then Privileged Mode Requested.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

substreamid_X bool
: If has a SubstreamID (!= ~0u) then Execution Requested (R must also be 1).

trans_id unsigned int
: The transaction ID that identifies this request.

untranslated_address unsigned int
: The Untranslated Address that this is requesting.

priqen_state

The trace of various fields that indicate the state of the PRIQ. Fields:

cons_incl_wrap unsigned int
: The value of SMMU_PRIQ.CONS.RD_and_RD_wrap.

number_of_pprs unsigned int
: The number of PPRs as indicated by the CONS/PROD.

number_of_pprs_still_to_deal_with unsigned int
: This is the number of PPRs that are currently waiting to either be written to the PRIQ, or auto-responded to.

number_of_priq_writes_in_flight unsigned int
: The number of writes to the PRIQ that are currently in flight.

ovackflg bool
: The OVACKFLG which if different to OVFLG is used to indicate that the PRIQ overflowed.

ovflg bool
: The OVFLG which if different to OVACKFLG is used to indicate that the PRIQ overflowed.

priq_abt_err bool
: There is an active SMMU_GERROR{N}.PRIQ_ABT_ERR.

priqen bool
: The value of SMMU_CR0.PRIQEN. The _effective_ value is 0 if SMMUEN == 0.

priqenack bool
: The value of SMMU_CR0ACK.PRIQEN.

prod_incl_wrap unsigned int
: The value of SMMU_PRIQ.CONS.WR_and_WR_wrap.

queue_disabled_due_to_prior_programming_error bool
: The queue was disabled as the programmer got CONS/PROD into an inconsistent state. The model will disable the PRIQ until SW disables and re-enables the queue via SMMU_CR0.PRIQEN.

smmuen bool
: The value of SMMU_CR0.SMMUEN. If this is 0 then the effective PRIQEN is 0.

smmuenack bool
: The value of SMMU_CR0.SMMUEN. If this is 0 then the effective PRIQEN is 0.

table_size_in_elements unsigned int
: The size of the table in the number of items it can hold.

ptw_read

Page Table Walk (read). This is the result of the physical access that the SMMU is making. Fields:

abort enum
: Non-zero if the access aborted/failed.

adomain enum
: The actual attributes of the access that was used, after IMP DEF mangling.

aprot enum
: The actual attributes of the access that was used, after IMP DEF mangling.

data unsigned int
: The data fetch if it didn't abort.

desckind enum
: The kind of descriptor

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

inner_cache enum
: The actual attributes of the access that was used, after IMP DEF mangling.

input_address unsigned int
: Input address this lookup is for (after TBI normalisation).

ns bool
: Non-Secure

outer_cache enum
: The actual attributes of the access that was used, after IMP DEF mangling.

pa_address unsigned int
: Physical address.

ssd_ns enum
: Security state of stream

streamid unsigned int
: The StreamID this is for.

substreamid unsigned int
: The SubstreamID this is for.

trans_id unsigned int
: Transaction id

ttb_grain_stage_and_level unsigned int
: Which TTB used in bits[31:24]. Which Grain size used in bit[23:16]:- 0 -- 4 KiB 1 -- 16 KiB 2 -- 64 KiB 3 -- 64 KiB ARMv8.2 where descriptor bits [15:12] for bits [51:48] of output address Stage in bits[15:8]. Level in bits[7:0].

ptw_read_st1_invalid_long_descriptor

Page Table Walk (read). The descriptor that the PTW fetched is invalid. Fields:

desckind enum
: The kind of descriptor

input_address unsigned int
: Input address this lookup is for (after TBI normalisation).

ns bool
: Non-Secure

pa_address unsigned int
: Physical address.

ssd_ns enum
: Security state of stream

streamid unsigned int
: The StreamID this is for.

substreamid unsigned int
: The SubstreamID this is for.

trans_id unsigned int
: Transaction id

ttb_grain_stage_and_level unsigned int
: Which TTB used in bits[31:24]. Which Grain size used in bit[23:16]:- 0 -- 4 KiB 1 -- 16 KiB 2 -- 64 KiB 3 -- 64 KiB ARMv8.2 where descriptor bits [15:12] for bits [51:48] of output address Stage in bits[15:8]. Level in bits[7:0].

ptw_read_st1_leaf_long_descriptor

Page Table Walk (read). The descriptor that the PTW fetched is a block or page and this is the decode. Fields:

AF bool
: Access Flag.

AP21 enum
: The access permissions.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

AttrIndx210 unsigned int
: The attribute index into the MAIR0/1.

DBM bool
: Dirty Bit Modifier. This only has meaning if running in AArch64 mode.

NS bool
: The encoding is for non-secure if this is a secure fetch.

PXN bool
: Privileged eXecute Never.

SH10 enum
: The shareability.

XN bool
: eXecute Never.

contiguous bool
: Contiguous hint.

desckind enum
: The kind of descriptor

hwu_pbha unsigned int
: Top four bits are appropriate CD.HWU*, bottom bits[62:59] of descriptor. Page Based Hardware Attributes: only valid on a bit-per-bit basis enabled by CD.HWU0*/CD.HWU1*.

input_address unsigned int
: Input address this lookup is for (after TBI normalisation).

nG bool
: not Global.

nT bool
: SMMUv3.2: If true then do not cache this entry in such a way that it can cause a TLB conflict abort (F_TLB_CONFLICT). The entry must produce a consistent result.

ns bool
: Non-Secure

output_address unsigned int
: Output address.

pa_address unsigned int
: Physical address.

ssd_ns enum
: Security state of stream

streamid unsigned int
: The StreamID this is for.

substreamid unsigned int
: The SubstreamID this is for.

trans_id unsigned int
: Transaction id

ttb_grain_stage_and_level unsigned int
: Which TTB used in bits[31:24]. Which Grain size used in bit[23:16]:- 0 -- 4 KiB 1 -- 16 KiB 2 -- 64 KiB 3 -- 64 KiB ARMv8.2 where descriptor bits [15:12] for bits [51:48] of output address Stage in bits[15:8]. Level in bits[7:0].

ptw_read_st1_table_long_descriptor

Page Table Walk (read). The descriptor that the PTW fetched is a Table, this decodes the fields. Fields:

APTable enum
: Remove permissions independently of subsequent descriptors.

NSTable bool
: The next level table descriptor is forced to non-secure.

PXNTable bool
: Force PXN independently of subsequent descriptors.

TableAddress unsigned int
: Address of the next table.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

XNTable bool
: Force XN independently of subsequent descriptors.

desckind enum
: The kind of descriptor

input_address unsigned int
: Input address this lookup is for (after TBI normalisation).

ns bool
: Non-Secure

pa_address unsigned int
: Physical address.

ssd_ns enum
: Security state of stream

streamid unsigned int
: The StreamID this is for.

substreamid unsigned int
: The SubstreamID this is for.

trans_id unsigned int
: Transaction id

ttb_grain_stage_and_level unsigned int
: Which TTB used in bits[31:24]. Which Grain size used in bit[23:16]:- 0 -- 4 KiB 1 -- 16 KiB 2 -- 64 KiB 3 -- 64 KiB ARMv8.2 where descriptor bits [15:12] for bits [51:48] of output address Stage in bits[15:8]. Level in bits[7:0].

ptw_read_st2_invalid_descriptor

Page Table Walk (read). The descriptor that the PTW fetched is invalid. Fields:

desckind enum
: The kind of descriptor

input_address unsigned int
: Input address this lookup is for (after TBI normalisation).

ns bool
: Non-Secure

pa_address unsigned int
: Physical address.

ssd_ns enum
: Security state of stream

streamid unsigned int
: The StreamID this is for.

substreamid unsigned int
: The SubstreamID this is for.

trans_id unsigned int
: Transaction id

ttb_grain_stage_and_level unsigned int
: Which TTB used in bits[31:24]. Which Grain size used in bit[23:16]:- 0 -- 4 KiB 1 -- 16 KiB 2 -- 64 KiB 3 -- 64 KiB ARMv8.2 where descriptor bits [15:12] for bits [51:48] of output address Stage in bits[15:8]. Level in bits[7:0].

ptw_read_st2_leaf_descriptor

Page Table Walk (read). The descriptor that the PTW fetched is a block or page and this is the decode. Fields:

AF bool
: Access Flag.

DBM bool
: Dirty Bit Modifier. This only has meaning if running in AArch64 mode.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

HAP21 enum
: The access permissions.

MemAttr3_0 enum
: The memory attributes.

SH10 enum
: The shareability.

XN enum
: eXecute Never.

contiguous bool
: Contiguous hint.

desckind enum
: The kind of descriptor

input_address unsigned int
: Input address this lookup is for (after TBI normalisation).

nT bool
: SMMUv3.2: If true then do not cache this entry in such a way that it can cause a TLB conflict abort (F_TLB_CONFLICT). The entry must produce a consistent result.

ns bool
: Non-Secure

output_address unsigned int
: Output address.

pa_address unsigned int
: Physical address.

s2hwu_pbha unsigned int
: Top four bits are STE.S2HWU*, bottom bits[62:59] of descriptor. Page Based Hardware Attributes: only valid on a bit-per-bit basis enabled by STE.S2HWU*.

ssd_ns enum
: Security state of stream

streamid unsigned int
: The StreamID this is for.

substreamid unsigned int
: The SubstreamID this is for.

trans_id unsigned int
: Transaction id

ttb_grain_stage_and_level unsigned int
: Which TTB used in bits[31:24]. Which Grain size used in bit[23:16]:- 0 -- 4 KiB 1 -- 16 KiB 2 -- 64 KiB 3 -- 64 KiB ARMv8.2 where descriptor bits [15:12] for bits [51:48] of output address Stage in bits[15:8]. Level in bits[7:0].

ptw_read_st2_table_descriptor

Page Table Walk (read). The descriptor that the PTW fetched is a Table, this decodes the fields. Fields:

APTable enum
: Remove permissions independently of subsequent descriptors.

PXNTable bool
: Force PXN independently of subsequent descriptors.

TableAddress unsigned int
: Address of the next table.

XNTable bool
: Force XN independently of subsequent descriptors.

desckind enum
: The kind of descriptor

input_address unsigned int
: Input address this lookup is for (after TBI normalisation).

ns bool
: Non-Secure

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

pa_address unsigned int
: Physical address.

ssd_ns enum
: Security state of stream

streamid unsigned int
: The StreamID this is for.

substreamid unsigned int
: The SubstreamID this is for.

trans_id unsigned int
: Transaction id

ttb_grain_stage_and_level unsigned int
: Which TTB used in bits[31:24]. Which Grain size used in bit[23:16]:- 0 -- 4 KiB 1 -- 16 KiB 2 -- 64 KiB 3 -- 64 KiB ARMv8.2 where descriptor bits [15:12] for bits [51:48] of output address Stage in bits[15:8]. Level in bits[7:0].

raw_register_end_read

The raw register read transaction. This is the transaction as directed to the register port. Fields:

id unsigned int
: Id of this transaction.

ok bool
: Was the access OK? The bus response will always be OK, but is the register access reasonable.

read_data unsigned int
: The data read.

raw_register_end_write

The raw register write transaction. This is the transaction as directed to the register port. Fields:

id unsigned int
: Id of this transaction.

ok bool
: Was the access OK? The bus response will always be OK, but is the register access reasonable.

raw_register_start_read

The raw register read transaction. This is the transaction as directed to the register port. Fields:

address unsigned int
: Address of transaction.

id unsigned int
: Id of this transaction. Top bit is set if is a debug transaction.

ns enum
: Is the transaction non-secure?

number_of_beats unsigned int
: Number of beats.

size_of_beat_in_bytes unsigned int
: Size of each beat in bytes.

raw_register_start_write

The raw register write transaction. This is the transaction as directed to the register port. Fields:

address unsigned int
: Address of transaction

id unsigned int
: Id of this transaction. Top bit is set if is a debug transaction.

ns enum
: Is the transaction non-secure?

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

number_of_beats unsigned int
: Number of beats.

size_of_beat_in_bytes unsigned int
: Size of each beat in bytes.

write_data unsigned int
: The data to write.

register_disallowed_read_string

A text representation of the read of a register that was disallowed. Fields:

out string
: The text description of the register value read.

register_disallowed_write_string

A text representation of the write of a register write that was disallowed. Fields:

in string
: The text description of the register value written.

register_read_reserved

A text representation of an access to a register address that is reserved. Fields:

in string
: The text description of the register value.

register_read_string

A text representation of the read of a register. Fields:

out string
: The text description of the register value read.

register_write_reserved

A text representation of an access to a register address that is reserved or a write.to a RES0 field in a register. Fields:

in string
: The text description of the register value.

register_write_string

A text representation of the write of a register. Fields:

in string
: The text description of the register value written.

reset_in

The reset signal. Fields:

value bool
: The value of the signal.

s_cmd_sync_completed_irq

Secure CMD_SYNC completed its IRQ action. Fields:

cmd_id unsigned int
: The ID of the command being issued.

cons_incl_wrap unsigned int
: CONS pointer of CMD_SYNC.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

s_cmd_sync_completed_no_action

Secure CMD_SYNC completed, there was no CMD_SYNC action (SIG_NONE). Fields:

cmd_id unsigned int
: The ID of the command being issued.

cons_incl_wrap unsigned int
: CONS pointer of CMD_SYNC.

s_cmd_sync_completed_sev

Secure CMD_SYNC completed its action of SIG_SEV. Fields:

cmd_id unsigned int
: The ID of the command being issued.

cons_incl_wrap unsigned int
: CONS pointer of CMD_SYNC.

s_cmd_sync_completed_with_error

Secure CMD_SYNC completed, however, there was an error associated with the completion of the CMD_SYNC. Fields:

cerror enum
: The error associated with the CMD_SYNC

cmd_id unsigned int
: The ID of the command being issued.

cons_incl_wrap unsigned int
: CONS pointer of CMD_SYNC.

s_cmd_sync_issued

Secure CMD_SYNC has now completed issuing and we now know if something is hazarding the CMD_SYNC from completing. Fields:

cmd_id unsigned int
: The ID of the command being issued.

cons_incl_wrap unsigned int
: CONS pointer of CMD_SYNC.

hazarded bool
: The CMD_SYNC is currently hazarded from completing.

s_cmd_sync_issuing

Secure CMD_SYNC is starting to issue, we are going to start accounting for which things have to be part of the prior set of things that have to complete before the CMD_SYNC can complete. Fields:

cmd_id unsigned int
: The ID of the command being issued.

cons_incl_wrap unsigned int
: CONS pointer of CMD_SYNC.

s_cmd_sync_starting_completion_action_irq

Secure CMD_SYNC starting IRQ completion action. Fields:

cmd_id unsigned int
: The ID of the command being issued.

cons_incl_wrap unsigned int
: CONS pointer of CMD_SYNC.

msiaddress unsigned int
: The MSI address to use (or 0 if not sending an MSI).

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

msidata unsigned int
The MSI data to send.

s_cmd_sync_starting_completion_action_sev

Secure CMD_SYNC starting its completion action of SIG_SEV. Fields:

cmd_id unsigned int
The ID of the command being issued.
cons_incl_wrap unsigned int
CONS pointer of CMD_SYNC.

s_cmdq_cmd_sync_error

The CMD_SYNC can fault if an ATC Invalidate timeout occurs. The timeout will be attached to the next CMD_SYNC on the queue on which the invalidate was launched. Fields:

cerror enum
The CERROR to attach.
cmd_queue unsigned int
The top bits indicate the CMDQ id. These bits are attached to all cmd_ids generated for this queue.

s_cmdq_cmd_sync_error_delivered

The CMD_SYNC can fault if an ATC Invalidate timeout occurs. This is the CMD_SYNC delivering that fault. Fields:

cerror enum
The CERROR to attach.
cmd_id unsigned int
Command ID of the CMD_SYNC

s_cmdq_ecmdq_enack

Trace the changes to ENACK. Fields:

cerror enum
The error being exposed, if any.
cmd_queue unsigned int
The top bits indicate the CMDQ id. These bits are attached to all cmd_ids generated for this queue.
enack bool
The value of ENACK.
forced_error bool
The queue was enabled with ERR != ERRACK and so the implementation chooses to trigger the error reporting mechanism.

s_cmdq_issue

A command is actually being executed. Fields:

cmd_id unsigned int
Command ID of the command being executed.
cons unsigned int
CONS of the command.
what string
What is being executed.

s_cmdq_pointers_state

The command queue pointers. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

cmd_queue unsigned int
: The top bits indicate the CMDQ id. These bits are attached to all cmd_ids generated for this queue.

disable_fetch bool
: The model has decided that as you programmed PROD/CONS inconsistently that it would not fetch as it has no confidence as to which are commands and which are not.

q_issuable_and_issued_cons unsigned int
: The CONS pointer of the region that is issuable and/or issued (including wrap).

q_issuable_and_issued_prod unsigned int
: The PROD pointer of the region that is issuable and/or issued (including wrap).

q_programmer_cons unsigned int
: The CONS pointer as seen by the programmer (including wrap).

q_programmer_prod unsigned int
: The PROD pointer as seen by the programmer (including wrap).

q_unfetched_cons unsigned int
: The CONS pointer of the region unfetched (including wrap).

q_unfetched_prod unsigned int
: The PROD pointer of the region unfetched (including wrap).

s_cmdq_state

A description of what the CMDQ can do now. Fields:

can_cerror bool
: There is a CERROR that can be recognised.

can_cmd_sync bool
: There is a CMD_SYNC that can be recognised.

can_fetch bool
: The SMMU can fetch

can_issue bool
: There are fetched commands that are eligible to be issued.

can_update_cons bool
: The programmer view of CMDQ_CONS can be moved to say some have been consumed.

cmd_queue unsigned int
: The top bits indicate the CMDQ id. These bits are attached to all cmd_ids generated for this queue.

cmd_sync enum
: Why a CMD_SYNC cannot be issued

cmdqen bool
: CMDQEN, when enabled then the CMDQ can fetch commands

cmdqen_1_to_0_unacknowledged bool
: CMDQEN has been set to 0, but we have yet to acknowledge it.

current-real-cerror enum
: The current CERROR that would be visible to the programmer if there was an error.

fetch-disabled_due_to_programmer_error bool
: The PROD/CONS pointer was inconsistent and the model decided to stop processing commands.

fetch-number_of_unfetched_commands unsigned int
: The number of commands that have yet to be fetched from memory, or ~0u if CMDQEN == 0.

fetch-state enum
: The state of the fetch state machine

fetch-too_many_commands_outstanding_so_not_fetching_any_more bool
: If there are too may issued commands, or pending to be issued then we won't fetch any more.

issue-number_of_issuable_commands unsigned int
: The number of commands that have been fetched but not yet issued.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

issue-number_of_unissuable_commands unsigned int
: This might include commands behind an illegal command, or behind a CMD_SYNC. Under some circumstances this can include a command representing an external abort.

issue-state enum
: The internal issued state machine state

outstanding_fetches_since_reset unsigned int
: The number of outstanding fetches.

s_eventq_adding_to_cmd_sync_prior_set

The event created by the specified transaction id must become visible to the programmer before the CMD_SYNC can complete. Fields:

trans_id unsigned int
: The transaction id that caused the event.

s_eventq_auto_retry_stalled_transaction

A transaction that was stalled but not yet reported on the Event Queue was retried. The retry can happen if the transaction didn't write because it was de-duplicated behind a reported stalling transaction and that stalling transaction was CMD_RESUMEd, or if the queue was unwriteable and the queue is now writeable. Fields:

trans_id unsigned int
: Transaction id of the auto-retried transaction.

why enum
: Why the transaction retried.

s_eventq_cmd_sync_unhazarded

The CMD_SYNC has been unhazarded as the prior set is empty and as much of it as required is visible in the programmers' view. Fields:

prod_incl_wrap unsigned int
: The prod index including wrap bit that was required to become visible.

prog_prod_incl_wrap unsigned int
: The current programmer-visible prod including wrap bit.

s_eventq_eventqen_and_eventqenack

Trace the values of EVENTQEN and EVENTQENACK as they change. Fields:

about enum
: Some more information about the queue state.

cons_incl_wrap unsigned int
: The CONS pointer (including wrap bits).

eventqen bool
: EVENTQEN.

eventqenack bool
: EVENTQENACK.

prod_incl_wrap unsigned int
: The PROD pointer (including wrap bits).

s_eventq_external_abort

A particular eventq record got an external abort. The event will subsequently appear in the ns/ s_eventq_lost_event_records trace. Fields:

prod unsigned int
: The prod pointer (excluding wrap bit) of the event record that aborted.

trans_id unsigned int
: The transaction id that produced the event record that aborted.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

s_eventq_lost_event_records

Event records that are lost and never record appear on this trace. Fields:

dword0 unsigned int
: Double word 0 of event.

dword1 unsigned int
: Double word 1 of event.

dword2 unsigned int
: Double word 2 of event.

dword3 unsigned int
: Double word 3 of event.

event enum
: Event code that was thrown away.

prod unsigned int
: The PROD index of the event if appropriate or ~0u otherwise.

reason enum
: The reason the record was lost.

trans_id unsigned int
: The transaction id that this corresponds to.

s_eventq_overflow_acknowledged

When events are lost then they appear on the eventq_lost_event_records_* trace. This will activate the overflow flag in the corresponding SMMU_s_EVENTQ_PROD register. Software can then acknowledge that flag using the overflow acknowledge flag in SMMU_s_EVENTQ_CONS. Fields:

bad bool
: If true then the programmer has attempted to acknowledge an overflow condition that didn't exist.

count_of_events_lost_due_to_overflow unsigned int
: This is the number of events that were lost because the event queue was full. This does not include any lost because of an EVENTQ_ABT_ERR.

s_eventq_pending_event_records

Event records that are pending to be dealt with by the eventq state machine. A specific event can be traced multiple times if it becomes part of the prior set of transactions of a CMD_SYNC that has to complete before the CMD_SYNC is allowed to be completed. Fields:

dword0 unsigned int
: Double word 0 of event.

dword1 unsigned int
: Double word 1 of event.

dword2 unsigned int
: Double word 2 of event.

dword3 unsigned int
: Double word 3 of event.

event enum
: Event code.

is_prior_set bool
: The transaction forms part of the prior set of a CMD_SYNC that have to become visible (or thrown away if necessary) before the CMD_SYNC can complete

trans_id unsigned int
: The transaction id that this corresponds to.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

s_eventq_prod_cons_updated

This records when the programmer visible PROD/CONS pointer is updated. More can be visible in memory than are advertised in the programmer-visible PROD pointer. This also traces the latest PROD that could have been advertised. Fields:

mem_prod unsigned int
: The PROD pointer value for all records that are visible in the memory.

num_events_available unsigned int
: The number of events available to the programmer.

prog_cons unsigned int
: The programmer-visible CONS pointer.

prog_prod unsigned int
: The programmer-visible PROD pointer.

what enum
: What changed?

s_eventq_raising_eventq_abt_err

EVENTQ_ABT_ERR can now be raised.

s_eventq_removing_from_cmd_sync_prior_set

The event captured by a CMD_SYNC and created by the specified transaction id is now in memory. Fields:

prod unsigned int
: The prod index (no wrap bit) that must become visible to the programmer to see this record. If not head_of_line then the record will not become visible and this field is meaningless.

reason enum
: The reason the event was removed.

trans_id unsigned int
: The transaction id that caused the event.

s_eventq_stashing_unreported_stalled_transaction_for_auto_retry

A stall event record would have been lost, instead it is stashed away to auto-retry when it can. Fields:

dword0 unsigned int
: Double word 0 of event.

dword1 unsigned int
: Double word 1 of event.

dword2 unsigned int
: Double word 2 of event.

dword3 unsigned int
: Double word 3 of event.

event enum
: Event code that was thrown away.

reason enum
: The reason the record was lost.

trans_id unsigned int
: The transaction id that this corresponds to.

s_eventq_write_event

An event is committed to being written. Fields:

dword0 unsigned int
: Double word 0 of event.

dword1 unsigned int
: Double word 1 of event.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

dword2 unsigned int
: Double word 2 of event.

dword3 unsigned int
: Double word 3 of event.

event enum
: Event code that was thrown away.

prod unsigned int
: The PROD pointer (without wrap) that this event will be written to.

stall_event bool
: Describes an event for a stalled transaction.

trans_id unsigned int
: Transaction ID that generated this event.

s_eventq_write_event_visible_in_memory

An event is now visible in memory. Fields:

abort bool
: The write aborted.

dword0 unsigned int
: Double word 0 of event.

dword1 unsigned int
: Double word 1 of event.

dword2 unsigned int
: Double word 2 of event.

dword3 unsigned int
: Double word 3 of event.

event enum
: Event code that was thrown away.

prod unsigned int
: The PROD pointer (without wrap) that this event will be written to.

stall_event bool
: Describes an event for a stalled transaction.

trans_id unsigned int
: Transaction ID that generated this event.

s_gerror_inverted

A GERROR bit was attempted to be inverted to record a fault. Fields:

already_different_to_gerrorn bool
: True if the bit was already different to the corresponding bit in GERRORN and so it was not inverted.

gerrorn unsigned int
: GERRORN register value.

interrupt_potentially_sent bool
: An interrupt is potentially sent, depending on IRQEN.
GERROR(N).MSI_GERROR_ABT_ERR does not send an interupt.

new_gerror unsigned int
: GERROR register value after any invertion.

which enum
: Which bit was attempted to be inverted.

s_gerrorn_acknowledge

A GERROR was acknowledged by SW. Fields:

which enum
: Which GERROR was acknowledged.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

sev

Send a SEV. Fields:

why enum
: Why the SEV was generated.

smmu_atc_inv

The CMD_ATC_INV command is sent. Fields:

address unsigned int
: Untranslated Address to invalidate.

cmd_id unsigned int
: Command id, if top-bit is set then was issued from the Non-secure CMDQ.

cons unsigned int
: CONS of the command.

global bool
: Global flag.

log2_size_in_bytes unsigned int
: Log 2 of the size in bytes of the region to invalidate.

ns_number_in_flight unsigned int
: Number of ATC invalidates in flight issued from the Non-secure CMDQ.

ns_number_waiting_to_go_out unsigned int
: The total number of ATC invalidates waiting to be pushed out from the Non-secure CMDQ.

s_number_in_flight unsigned int
: Number of ATC invalidates in flight issued from the Secure CMDQ.

s_number_waiting_to_go_out unsigned int
: The total number of ATC invalidates waiting to be pushed out from the Secure CMDQ.

streamid unsigned int
: StreamID in the invalidate command.

substreamid unsigned int
: SubstreamID or ~0u if SSValid == false.

smmu_atc_inv_completed

The CMD_ATC_INV command completed. Fields:

cmd_id unsigned int
: Command id, if top-bit is set then was issued from the Non-secure CMDQ.

ns_number_in_flight unsigned int
: Number of ATC invalidates in flight now this one has completed, that were issued by the Non-secure CMDQ.

ns_number_waiting_to_go_out unsigned int
: The total number of ATC invalidates waiting to be pushed out from the Non-secure CMDQ.

response enum
: The response.

s_number_in_flight unsigned int
: Number of ATC invalidates in flight now this one has completed, that were issued by the Secure CMDQ.

s_number_waiting_to_go_out unsigned int
: The total number of ATC invalidates waiting to be pushed out from the Secure CMDQ.

smmu_atc_inv_end

An ATC invalidate request has been started. Fields:

PASID_global bool
: If has a SubstreamID (PASID) then if it is 'global' or not.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

address unsigned int
: The Untranslated Address to use.

log2_size_in_bytes unsigned int
: Log2 of the size of the region in bytes to invalidate.

response enum
: The response to the ATC invalidate

streamid unsigned int
: The StreamID to use.

substreamid unsigned int
: The SubstreamID to use (or ~0u if not used). This corresponds to the PASID.

smmu_atc_inv_start

An ATC invalidate request has been started. Fields:

PASID_global bool
: If has a SubstreamID (PASID) then if it is 'global' or not.

address unsigned int
: The Untranslated Address to use.

log2_size_in_bytes unsigned int
: Log2 of the size of the region in bytes to invalidate.

streamid unsigned int
: The StreamID to use.

substreamid unsigned int
: The SubstreamID to use (or ~0u if not used). This corresponds to the PASID.

smmu_ats_initial

This is the initial ATS request. Fields:

ia unsigned int
: Input address

max_number_of_replies unsigned int
: The maximum number of replies allowed to return

no_write bool
: The NW (no write flag) of the ATS request. If clear then the requester is going to do a write.

pasid_execute_requested bool
: If there is a PASID (substreamid != ~0u) then this represents the requester asking for execution permissions.

pasid_privileged_mode_requested bool
: If there is a PASID (substreamid != ~0u) then this represents the requester asking for the privileged mode's permissions.

ssd_ns bool
: Incoming SSD, only non-secure ATS requests are legal.

streamid unsigned int
: StreamID of the ATS request.

substreamid unsigned int
: SubstreamID (which is identical to the PASID) of the ATS request. If no PASID-prefix is sent then this is ~0u

tbu unsigned int
: Translation Buffer Unit number

smmu_ats_reply_failure

This is an ATS reply indicating failure. Fields:

event enum
: Equivalent event number that would have been generated for an equivalent ordinary transaction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

failure enum
: What is the failure response code?

state enum
: The transaction state of the successfully ATS request.

smmu_ats_reply_success

This is an ATS reply, typically the SMMU will only return a single response, even if the requester indicated it could accept more replies. NOTE that the SMMU responds with 'success' in some cases when a fault is encountered and RW==0. Fields:

N bool
: Non-snooped access. If one then the requester must clear the NoSnoop bit on transactions, unless otherwise enabled in a Function-specific manner.

P bool
: Privileged mode. These permissions related to privileged mode.

RWX enum
: Read/Write/Execute

U bool
: Untranslated access. If one, and RW !=0 then use UntranslatedAccesses for the allowed accesses by RW(X).

inner enum
: The inner cacheability attributes to use for TranslatedAccesses

input_address unsigned int
: Input address of the ATS request

instcfg enum
: The STE.INSTCFG field

outer enum
: The outer cacheability attributes to use for TranslatedAccesses

privcfg enum
: The STE.PRIVCFG field

shareability unsigned int
: The shareability to use for TranslatedAccesses

size unsigned int
: The size of the region covered by this translation.

state enum
: The transaction state of the successfully ATS request.

translated_address unsigned int
: If RW!=0 && U != 0, then the Translated Address that a TranslatedAccess can be made with

smmu_final_transaction

This is the transaction group request to remap has completed one set of remapping. For ordinary transactions, this represents a bundle of transactions with the same attributes but different addresses within a certain range around the address. If it stalls then it will report through this trace source, stall (stag_if_stalling != ~0u) and when resume will issue another smmu_initial_transaction as it undergoes remapping again. This trace source can also represent part of the process for ATOS/ATS or finding STE.PPAR for PRI requests that need to be auto-responded to. Fields:

axmmuflow enum
: The AxMMUFLOW for this transaction group. stallable -- allow the transaction to be stalled if configured. translated-access -- is a PCIe Translated Access non-stallable -- do not stall transaction even if configured. transfault-flow -- do not report for Translation Related Faults, but tell client device to use a page request mechanism instead (for example PRI).

cmo_point enum
: The point associated with the CMO, if applicable

input_address unsigned int
: The input address of the transaction group.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

input_inner enum
: Inner cacheability for the input attributes.

input_outer enum
: Outer cacheability for the input attributes.

input_prot enum
: The instruction/data, bus-ns/s and privileged/user marking

input_sh enum
: Shareability for the input attributes.

ipa_address unsigned int
: The IPA of the transaction.

kind enum
: The kind of access this transaction group that this represents.

mpam_pmg_and_partid unsigned int
: Bits[23:16] are the MPAM PMG. Bits[15:0] are the physical PARTID.

operation enum
: The kind of operation that this represents.

output_address unsigned int
: The input address of the transaction group.

output_inner enum
: Inner cacheability for the output attributes.

output_outer enum
: Outer cacheability for the output attributes.

output_prot enum
: The instruction/data, bus-ns/s and privileged/user marking

output_sh enum
: Shareability for the output attributes.

output_vmid unsigned int
: The output VMID/GBPA.IMPDEF or ~0u if not valid.

size_of_region_in_bytes unsigned int
: An imp def size of region for which this translation is valid for.

ssd_ns enum
: The security state of the transaction

stag_if_stalling unsigned int
: This is the STAG used by the transaction if it is going to stall. It is ~0u if it is not going to stall.

state enum
: The final transaction state.

streamid unsigned int
: The StreamID of the transsction.

substreamid unsigned int
: The SubstreamID of the transaction or 0xFFFFffff if no SubstreamID.

tbu unsigned int
: Translation Buffer Unit number.

trans_id unsigned int
: Transaction id. Top bit set if debug transaction.

smmu_initial_transaction

This is the transaction group request to remap is going to start one set of remapping. For ordinary transactions, this represents a bundle of transactions with the same attributes but different addresses within a certain range around the address. A stalling transaction will report through this trace source when it unstalls. This trace source can also represent part of the process for ATOS/ATS or finding STE.PPAR for PRI requests that need to be auto-responded to. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

axmmuflow enum
: The AxMMUFLOW for this transaction group. stallable -- allow the transaction to be stalled if configured. translated-access -- is a PCIe Translated Access non-stallable -- do not stall transaction even if configured. transfault-flow -- do not report for Translation Related Faults, but tell client device to use a page request mechanism instead (for example PRI).

cmo_point enum
: The point associated with the CMO, if applicable

input_address unsigned int
: The input address of the transaction group.

input_inner enum
: Inner cacheability for the input attributes.

input_outer enum
: Outer cacheability for the input attributes.

input_prot enum
: The instruction/data, bus-ns/s and privileged/user marking

input_sh enum
: Shareability for the input attributes.

kind enum
: The kind of access this transaction group that this represents.

operation enum
: The kind of operation that this represents.

ssd_ns enum
: The security state of the transaction

streamid unsigned int
: The StreamID of the transcton.

substreamid unsigned int
: The SubstreamID of the transaction or 0xFFFFffff if no SubstreamID.

tbu unsigned int
: Translation Buffer Unit number.

trans_id unsigned int
: Transaction id. Top bit set if debug transaction.

smmu_poison_tw_data

Poison data has been returned to a table walk transaction. Fields:

bitmap_of_poison unsigned int
: The bitmap of which beats of the transaction where poisoned.

is_cas bool
: True if this is a compare-and-swap operation.

ns bool
: Non-secure state of the bus transaction.

number_of_64bit_beats unsigned int
: Number of beats this transaction fetched.

paddress unsigned int
: Physical address of the table walk transaction.

ras_group_id unsigned int
: If non-~0u then is the RAS group that we wish to attach the error to. Otherwise will use the IMP DEF policy.

ras_record_index_in_group unsigned int
: If non-~0u then is the RAS record index in the group that we wish to attach the error to. Otherwise will use the IMP DEF policy.

what enum
: What table walk was being performed.

smmu_pri_resp

The CMD_PRI_RESP command is queued to be sent to the PCIe system. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

cmd_id unsigned int
: Command id

cons unsigned int
: CONS of the command.

prgindex unsigned int
: PRG Index.

response enum
: If has a SubstreamID (PASID) then if it is 'global' or not.

streamid unsigned int
: StreamID in the invalidate command.

substreamid unsigned int
: SubstreamID or ~0u if SSValid == false.

smmu_pri_resp_nop

The CMD_PRI_RESP command was NOPped. Fields:

cmd_id unsigned int
: Command id

cons unsigned int
: CONS of the command.

prgindex unsigned int
: PRG Index.

response enum
: If has a SubstreamID (PASID) then if it is 'global' or not.

streamid unsigned int
: StreamID in the invalidate command.

substreamid unsigned int
: SubstreamID or ~0u if SSValid == false.

why enum
: Why the CMD_PRI_RESP was NOPped.

smmu_priq_resp_fake_return

A PRIQ Response is posted to the PCIe subsystem and so has no acknowledgement that it is received. However, in the model then we artificially know when the the PRIQ Response has been delivered to the PCIe subsystem, even if the ATC has not yet acted on it. Fields:

prgindex unsigned int
: The PRG Index this request corresponds to.

streamid unsigned int
: The StreamID to use.

substreamid unsigned int
: The SubstreamID to use (or ~0u if not used). This corresponds to the PASID.

what enum
: What happened.

smmu_priq_resp_start

A PRIQ Response has been posted to the PCIe subsystem. As the response is posted then there is no way of knowing when it is received by the EndPoint. Fields:

prgindex unsigned int
: The PRG Index this request corresponds to.

response enum
: If has a SubstreamID (PASID) then if it is 'global' or not.

streamid unsigned int
: The StreamID to use.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

substreamid unsigned int
: The SubstreamID to use (or ~0u if not used). This corresponds to the PASID.

smmu_thread_wait_wake

Traces a thread's wait/wake status. Fields:

current_ticks unsigned int
: The current tick count of simulated time.

event enum
: What is happening to this thread.

thread_index unsigned int
: The ID of this thread.

ticks unsigned int
: If the event relates to a time then this is held in this field. Otherwise, 0.

stall_transaction

A transaction is about to stall. Fields:

stag unsigned int
: STAG.

streamid unsigned int
: StreamID.

streamid_ns bool
: The StreamID was for the non-secure world.

trans_id unsigned int
: Transaction id.

stall_transaction_inhibited_by_STALL_MAX

A transaction is about to stall but the maximum number of transactions have stalled and we can't report this one to the event queue (even if non-full). Fields:

streamid unsigned int
: StreamID.

streamid_ns bool
: The StreamID was for the non-secure world.

trans_id unsigned int
: Transaction id.

stall_transaction_resuming

A stalled transaction is resuming. Fields:

stag unsigned int
: STAG if appropriate, or if was inhibited by STALL_MAX then 0xFAFA.

stallresult enum
: What the transaction resumed to do.

streamid unsigned int
: StreamID.

streamid_ns bool
: The StreamID was for the non-secure world.

trans_id unsigned int
: Transaction id.

start_ptw_read

Page Table Walk (read). This is the start of the physical access that the SMMU is making. Fields:

adomain enum
: The actual attributes of the access that was used, after IMP DEF mangling.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

aprot enum
: The actual attributes of the access that was used, after IMP DEF mangling.

desckind enum
: The kind of descriptor

inner_cache enum
: The actual attributes of the access that was used, after IMP DEF mangling.

input_address unsigned int
: Input address this lookup is for (after TBI normalisation).

mpam_pmg_and_partid unsigned int
: Bits[23:16] are the MPAM PMG. Bits[15:0] are the physical PARTID.

ns bool
: Non-Secure

outer_cache enum
: The actual attributes of the access that was used, after IMP DEF mangling.

pa_address unsigned int
: Physical address.

ssd_ns enum
: Security state of stream

streamid unsigned int
: The StreamID this is for.

substreamid unsigned int
: The SubstreamID this is for.

trans_id unsigned int
: Transaction id

ttb_grain_stage_and_level unsigned int
: Which TTB used in bits[31:24]. Which Grain size used in bit[23:16]:- 0 -- 4 KiB 1 -- 16 KiB 2 -- 64 KiB 3 -- 64 KiB ARMv8.2 where descriptor bits [15:12] for bits [51:48] of output address Stage in bits[15:8]. Level in bits[7:0].

ste_cc.STE_commentary

A commentary from the cache about what is going on. Fields:

output string
: The stream output.

ste_cc.STE_entry_invalidated

An entry in the cache has been invalidated. Fields:

entry_id unsigned int
: The entry id that is being invalidated.

reason enum
: Why the entry is being invalidated.

ste_entry_allocated

An STE entry has been allocated. Fields:

entry_id unsigned int
: The id of the cache entry. Top bit is set if is a debug entry.

ns enum
: For the non-secure world

ste string
: A textual description of the STE.

streamid unsigned int
: The StreamID of the entry this will match.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

tlb_entry_allocated

A TLB entry has been allocated. Fields:

aset unsigned int
: The ASET of this entry.

asid unsigned int
: ASID if appropriate.

index unsigned int
: Index of TLB entry.

input_end_incl_address unsigned int
: The end inclusive address of the input range that this matches.

input_start_address unsigned int
: The start address of the input range that this matches.

output_end_incl_address unsigned int
: The end inclusive address of the output range.

output_start_address unsigned int
: The start address of the output range.

scheme enum
: The tagging scheme used

ssd_ns enum
: Security State

tbi bool
: Was the entry formed using Top Byte Ignore (TBI).

vmid unsigned int
: VMID if appropriate.

tlb_info_tlb_entries_overlap

A TLB entry was inserted into the TLB and it overlaps an existing entry. This isn't a problem as it was inserted in such a way that it architecturally works. Fields:

end_address_of_new_entry unsigned int
: End address of new entry.

end_address_of_old_entry unsigned int
: End address of old entry.

how_inserted enum
: How the entry was inserted.

index_of_new_entry unsigned int
: Index of new entry.

index_of_old_entry unsigned int
: Index of old entry.

start_address_of_new_entry unsigned int
: Start address of new entry.

start_address_of_old_entry unsigned int
: Start address of old entry.

verbose_commentary

This is a verbose commentary on the translation process the SMMU is performing. Fields:

output string
: The stream output.

warning_MSI_EVENTQ_address_out_of_range_of_oas

The MSI address is configured outside of the range of the downstream address bus size. Fields:

address unsigned int
: The untruncated address of the MSI.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

warning_MSI_GERROR_address_out_of_range_of_oas

The MSI address is configured outside of the range of the downstream address bus size. Fields:

address unsigned int
: The untruncated address of the MSI.

warning_MSI_PRIQ_address_out_of_range_of_oas

The MSI address is configured outside of the range of the downstream address bus size. Fields:

address unsigned int
: The untruncated address of the MSI.

warning_MSI_S_EVENTQ_address_out_of_range_of_oas

The MSI address is configured outside of the range of the downstream address bus size. Fields:

address unsigned int
: The untruncated address of the MSI.

warning_MSI_S_GERROR_address_out_of_range_of_oas

The MSI address is configured outside of the range of the downstream address bus size. Fields:

address unsigned int
: The untruncated address of the MSI.

warning_MSI_pmcg_address_out_of_range_of_oas

The MSI Address of the Performance Monitor Counter Group (PMCG) is out of range of the OAS and so will be silently truncated. Fields:

address unsigned int
: The untruncated address of the MSI.

pmcg_index unsigned int
: Index of the PMCG.

ssd_ns bool
: True if this PMCG controlled by the non-secure world.

warning_discarding_interrupt_EVENTQ_as_irqen_low

Interrupt generation is turned off by SMMU_IRQ_CTRL.EVENTQ_IRQEN.

warning_discarding_interrupt_GERROR_as_irqen_low

Interrupt generation is turned off by SMMU_IRQ_CTRL.GERROR_IRQEN.

warning_discarding_interrupt_PRIQ_as_irqen_low

Interrupt generation is turned off by SMMU_S_IRQ_CTRL.PRIQ_IRQEN.

warning_discarding_interrupt_S_EVENTQ_as_irqen_low

Interrupt generation is turned off by SMMU_S_IRQ_CTRL.EVENTQ_IRQEN.

warning_discarding_interrupt_S_GERROR_as_irqen_low

Interrupt generation is turned off by SMMU_S_IRQ_CTRL.GERROR_IRQEN.

warning_gerrorn_bad_acknowledge

A GERROR was acknowledged by SW in the GERRORN register that did not have an active error. The result is UNPREDICTABLE if this will generate an interrupt or not. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

which enum
: Which GERROR was acknowledge when there was no active error.

warning_reg_after_doesnt_match_written_value

A write occurred that tried to set bits in a register, that for one reason or another, failed to get written. Fields:

desc string
: The textual description of what happened.

warning_s_gerrorn_bad_acknowledge

A GERROR was acknowledged by SW in the GERRORN register that did not have an active error. The result is UNPREDICTABLE if this will generate an interrupt or not. Fields:

which enum
: Which GERROR was acknowledge when there was no active error.

what_going_to_do_with_terminated_event

A terminating transaction has produced an event, this tells you what the model is going to do with the event. Fields:

CD.S bool
: The CD.S field if available.

S2 bool
: The event is related to Stage 2.

STE.S1STALLD bool
: The STE.S1STALLD field if available.

STE.S2S bool
: The STE.S2S field if available.

aborts bool
: The transaction will abort.

axmmuflow enum
: The AxMMUFLOW for this transaction group. stallable -- allow the transaction to be stalled if configured. translated-access -- is a PCIe Translated Access non-stallable -- do not stall transaction even if configured. transfault-flow -- do not report for Translation Related Faults, but tell client device to use a page request mechanism instead (for example PRI).

is_tr_fault bool
: Is a Translation Related fault.

reports bool
: The transaction will attempt to report.

ssd_ns bool
: The transaction is classified as SSD non-secure.

supports_stall_model bool
: The implementation supports the stall model.

trans_id unsigned int
: The transaction id.

why_abort_decision enum
: The reason why the transaction aborted/did not abort.

why_report_decision enum
: The reason why the transaction reported/did not report.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.112 SMMUv3TestEngine - trace

This section describes the trace sources.

access_denied_due_to_security

The access was denied as the frame and the security PCTRL.SSD_NS did not allow access. Fields:

address unsigned int
: Address.

frame_in_set unsigned int
: Which frame in the 64 KiB set it belongs to.

is_read bool
: Is the transaction a read.

ns bool
: The NS of the transaction.

priv bool
: Is the set the privileged set of frames?

set_pair unsigned int
: Which 128 KiB set-pair it belongs to.

ssd_ns bool
: The SSD NS of the frame.

frame_misconfigured_trace

The frame was written but is misconfigured. Fields:

cmd unsigned int
: The command that is running.

frame_in_set unsigned int
: Which frame in the 64 KiB set it belongs to.

priv bool
: Is the set the privileged set of frames?

set_pair unsigned int
: Which 128 KiB set-pair it belongs to.

why enum
: Why it is misconfigured?

why_data unsigned int
: Data related to why it is misconfigured:- src-attributes-illegal : the halfword of the attributes dest-attributes-illegal : the halfword of the attributes msi-attributes-illegal : the halfword of the attributes downstream-port-index-out-of-range : port index substreamid-out-of-range : SubstreamID SUM64, begin is not aligned to uint64_t : begin SUM64, end_incl is not aligned to uint64_t : end_incl PCIe MSIAddress field should be 0/1 : msiaddress PCIe MSIData field should be < 2048 : msidata range-* : the end_incl

msi_aborted

An MSI aborted. Fields:

frame unsigned int
: Frame generating MSI.

msiaddress_or_pcie_msi_enable unsigned int
: MSI address, or in PCIe enable MSI

msiattr unsigned int
: MSI attributes. Ignored for PCIe

msidata_or_pcie_msi_index unsigned int
: MSI data, or in PCIe the MSI index to use

pcie_result enum
: If this MSI was generated on PCIe then the result

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

streamid unsigned int
: StreamID.

substreamid unsigned int
: SubstreamID.

msi_generated

An MSI is generated for the specified work load. Fields:

frame unsigned int
: Frame generating MSI.

msiaddress unsigned int
: MSI address.

msiattr unsigned int
: MSI attributes.

msidata unsigned int
: MSI data.

streamid unsigned int
: StreamID.

substreamid unsigned int
: SubstreamID.

read_access

A successful read access occurred. Fields:

address unsigned int
: Address

data unsigned int
: The data that was read.

frame_in_set unsigned int
: Which frame in the 64 KiB set it belongs to.

ns bool
: The NS of the transaction.

offset unsigned int
: Offset of the first byte read.

priv bool
: Is the set the privileged set of frames?

set_pair unsigned int
: Which 128 KiB set-pair it belongs to.

work_completed

The specified work was completed. Fields:

frame unsigned int
: Frame that completed.

work_memcpy

A frame has been setup to do memcpy. Fields:

attributes unsigned int
: Attributes.

begin unsigned int
: begin address.

dest unsigned int
: Destination address.

end_incl unsigned int
: End inclusive address.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

frame unsigned int
: Frame setup.

msiaddress unsigned int
: The msiaddress field.

msiattr unsigned int
: The msiattr field.

msidata unsigned int
: The msidata field.

seed unsigned int
: Seed.

streamid unsigned int
: The streamid to use if controllable.

stride unsigned int
: Stride.

substreamid unsigned int
: The substreamid to use if controllable.

work_rand48

A frame has been setup to do rand48. Fields:

attributes unsigned int
: Attributes.

begin unsigned int
: begin address.

end_incl unsigned int
: End inclusive address.

frame unsigned int
: Frame setup.

msiaddress unsigned int
: The msiaddress field.

msiattr unsigned int
: The msiattr field.

msidata unsigned int
: The msidata field.

seed unsigned int
: Seed.

streamid unsigned int
: The streamid to use if controllable.

stride unsigned int
: Stride.

substreamid unsigned int
: The substreamid to use if controllable.

work_sum64

A frame has been setup to do sum64. Fields:

attributes unsigned int
: Attributes.

begin unsigned int
: begin address.

end_incl unsigned int
: End inclusive address.

frame unsigned int
: Frame setup.

msiaddress unsigned int
: The msiaddress field.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

msiattr unsigned int
: The msiattr field.

msidata unsigned int
: The msidata field.

seed unsigned int
: Seed.

start_of_sum unsigned int
: The value of udata[1] that is the initial value that we start adding to.

streamid unsigned int
: The streamid to use if controllable.

stride unsigned int
: Stride.

substreamid unsigned int
: The substreamid to use if controllable.

work_unit_memcpy_finished

Part (or all) of the memcpy engine has finished a work unit. Fields:

attr unsigned int
: Source/Destination attributes from the frame.

dest_addr unsigned int
: Destination address.

frame unsigned int
: Frame doing work.

number_of_bytes unsigned int
: Number of bytes for this work unit.

ok bool
: Transfer was OK.

src_addr unsigned int
: Source address.

work_unit_memcpy_started

Part (or all) of the memcpy engine has started a work unit. Fields:

attr unsigned int
: Source/Destination attributes from the frame.

dest_addr unsigned int
: Destination address.

frame unsigned int
: Frame doing work.

number_of_bytes unsigned int
: Number of bytes for this work unit.

src_addr unsigned int
: Source address.

work_unit_rand48_finished

Part (or all) of the rand48 engine has finished a work unit. Fields:

attr unsigned int
: Source/Destination attributes from the frame.

dest_addr unsigned int
: Source address.

frame unsigned int
: Frame doing work.

number_of_bytes unsigned int
: Number of bytes for this work unit.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ok bool
: Transfer was OK.

seed_for_work_unit unsigned int
: The seed used for the work unit.

work_unit_rand48_started

Part (or all) of the rand48 engine has started a work unit. Fields:

attr unsigned int
: Source/Destination attributes from the frame.

dest_addr unsigned int
: Destination address.

frame unsigned int
: Frame doing work.

number_of_bytes unsigned int
: Number of bytes for this work unit.

seed_for_work_unit unsigned int
: The seed used for the work unit.

work_unit_sum64_finished

Part (or all) of the sum64 engine has finished a work unit. Fields:

attr unsigned int
: Source/Destination attributes from the frame.

frame unsigned int
: Frame doing work.

number_of_bytes unsigned int
: Number of bytes for this work unit.

ok bool
: Transfer was OK.

running_sum unsigned int
: The current running sum.

src_addr unsigned int
: Source address.

work_unit_sum64_started

Part (or all) of the sum64 engine has started a work unit. Fields:

attr unsigned int
: Source/Destination attributes from the frame.

frame unsigned int
: Frame doing work.

number_of_bytes unsigned int
: Number of bytes for this work unit.

src_addr unsigned int
: Source address.

write_access

A successful write access occurred. Fields:

address unsigned int
: Address

data unsigned int
: The data that was read.

frame_in_set unsigned int
: Which frame in the 64 KiB set it belongs to.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ns bool
: The NS of the transaction.

offset unsigned int
: Offset of the first byte read.

priv bool
: Is the set the privileged set of frames?

set_pair unsigned int
: Which 128 KiB set-pair it belongs to.

write_denied_as_running

The write was denied as the frame is running. Fields:

address unsigned int
: Address.

cmd unsigned int
: The command that is running.

frame_in_set unsigned int
: Which frame in the 64 KiB set it belongs to

priv bool
: Is the set the privileged set of frames?

set_pair unsigned int
: Which 128 KiB set-pair it belongs to

write_to_PCTRL_SSD_NS_ignored_trace

The write to PCTRL.SSD_NS was ignored as it was a non-secure transaction trying to change the frame to secure. The rest of the write to PCTRL has taken effect. Fields:

address unsigned int
: Address.

frame_in_set unsigned int
: Which frame in the 64 KiB set it belongs to

set_pair unsigned int
: Which 128 KiB set-pair it belongs to

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.113 SignalLogger - trace

This section describes the trace sources.

SIGNAL

Input signal to SignalLogger driven. Fields:

Value enum
: Value of signal being driven

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.114 TLB - trace

This section describes the trace sources.

MMU_TLB_CONFLICT

TLB conflict. Fields:

ASID unsigned int
: The Address Space Identifier of this TLB entry.

CORE_NUM unsigned int
: Core number in a multi processor.

Hyp bool
: Entry matches in Hyp state only

NS enum
: Secure state in which TLB Entry will match

PAGESIZE unsigned int
: Size of the region (log2)

PBASE unsigned int
: Physical base address of the region.

REGIME_EL unsigned int
: Entry matches in this translation regime

VBASE unsigned int
: Virtual base address of the region.

VMID unsigned int
: Virtual Machine Identifier

nG enum
: Flag indicating whether ASID will be matched

MMU_TLB_EVICT

TLB evict. Fields:

ASID unsigned int
: The Address Space Identifier of this TLB entry.

CORE_NUM unsigned int
: Core number in a multi processor.

Hyp bool
: Entry matches in Hyp state only

NS enum
: Secure state in which TLB Entry will match

PAGESIZE unsigned int
: Size of the region (log2)

PBASE unsigned int
: Physical base address of the region.

REGIME_EL unsigned int
: Entry matches in this translation regime

VBASE unsigned int
: Virtual base address of the region.

VMID unsigned int
: Virtual Machine Identifier

nG enum
: Flag indicating whether ASID will be matched

MMU_TLB_FILL

TLB fill. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ASID unsigned int
: The Address Space Identifier of this TLB entry.

ATA_PRIVILEGED bool
: Is the allocation tag access enabled for privileged accesses

ATA_UNPRIVILEGED bool
: Is the allocation tag access enabled for unprivileged accesses

CORE_NUM unsigned int
: Core number in a multi processor.

EPAN bool
: Enhanced PAN cached value for this page

Hyp bool
: Entry matches in Hyp state only

INNERCACHE_RA bool
: Is the inner cache allocate on read

INNERCACHE_TYPE enum
: Inner Caching scheme (NC/MB/WA).

INNERCACHE_WA bool
: Is the inner cache allocate on write

MEMTYPE enum
: Memory type.

NS enum
: Secure state in which TLB Entry will match

NSDesc enum
: Secure state of transactions made using the TLB Entry

OUTERCACHE_RA bool
: Is the outer cache allocate on read

OUTERCACHE_TYPE enum
: Outer Caching scheme (NC/MB/WA).

OUTERCACHE_WA bool
: Is the outer cache allocate on write

PAGESIZE unsigned int
: Size of the region (log2)

PBASE unsigned int
: Physical base address of the region.

PXN bool
: Privileged Execute Never

REGIME_EL unsigned int
: Entry matches in this translation regime

SH enum
: Shareability

SIDE enum
: Inst / Data.

TCMA bool
: Is the relevant TCMA/TCMA0/TCMA1 bit enabled

VBASE unsigned int
: Virtual base address of the region.

VMID unsigned int
: Virtual Machine Identifier

X16HINT bool
: 16 Entry Contiguous Hint

XN bool
: Execute Never

XS bool
: Limited TLBI XS value for this page

nG enum
: Flag indicating whether ASID will be matched

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

MMU_TLB_FLUSH

TLB flush.

MMU_TLB_FLUSH_ADDR

TLB flush, match address. Fields:

ADDR unsigned int
: Address

NS bool
: Is Non-Secure

NSHYP bool
: Is Non-Secure HYP

MMU_TLB_FLUSH_ADDR_RANGE

TLB flush, match address range. Fields:

ADDR_FIRST unsigned int
: First address in range

ADDR_LAST unsigned int
: Last address in range

NS bool
: Is Non-Secure

NSHYP bool
: Is Non-Secure HYP

MMU_TLB_FLUSH_ALL

TLB flush all entries.

MMU_TLB_FLUSH_ASID

TLB flush, match ASID. Fields:

ASID unsigned int
: ASID of TLB flush

MMU_TLB_FLUSH_VMID

TLB flush, match VMID. Fields:

VMID unsigned int
: VMID of TLB flush

MMU_TLB_HIT

TLB accesses hit. Fields:

ASID unsigned int
: The Address Space Identifier of this TLB entry.

CORE_NUM unsigned int
: Core number in a multi processor.

Hyp bool
: Entry matches in Hyp state only

NS enum
: Secure state in which TLB Entry will match

REGIME_EL unsigned int
: Entry matches in this translation regime

SIDE enum
: Inst / Data.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VADDR unsigned int
: Virtual address of the access.

VMID unsigned int
: Virtual Machine Identifier

MMU_TLB_MISS

TLB access miss. Fields:

ASID unsigned int
: The Address Space Identifier of this TLB entry.

CORE_NUM unsigned int
: Core number in a multi processor.

Hyp bool
: Entry matches in Hyp state only

NS enum
: Secure state in which TLB Entry will match

REGIME_EL unsigned int
: Entry matches in this translation regime

SIDE enum
: Inst / Data.

VADDR unsigned int
: Virtual address of the access.

VMID unsigned int
: Virtual Machine Identifier

MMU_TLB_SPILL

TLB removals caused by fill to a occupied slot.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.115 V550 - trace

This section describes the trace sources.

ArchMsg.Info.IgnoredRead

Read to reserved area attempted. Fields:

NSECURE bool
: True when the access is not a secure access

OFFSET unsigned int
: Access offset

ArchMsg.Info.IgnoredWrite

Write to reserved area attempted. Fields:

DATA unsigned int
: The data written to the address

NSECURE bool
: True when the access is not a secure access

OFFSET unsigned int
: Access offset

ArchMsg.Info.JobqueueUpdated

Job queue updated. Fields:

NEWVAL unsigned int
: New JOBQUEUE value

OLDVAL unsigned int
: Previous JOBQUEUE value

ArchMsg.Info.LSID0.Firmware.CommandFailure

Command failed. Fields:

CMD unsigned int
: The OMX Command that failed.

ERR unsigned int
: The reported error code.

ArchMsg.Info.LSID0.Firmware.InputBufferReceived

Received input buffer.

ArchMsg.Info.LSID0.Firmware.InvalidBufferHeader

Received invalid buffer header, cannot convert.

ArchMsg.Info.LSID0.Firmware.InvalidCallback

Invalid callback. Fields:

REASON string
: Reason for the invalidity of the callback.

ArchMsg.Info.LSID0.Firmware.OutputBufferReceived

Received output buffer.

ArchMsg.Info.LSID0.Firmware.UnexpectedCallback

Unexpected callback. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

REASON string
Reason for the invalidity of the callback.

ArchMsg.Info.LSID0.Firmware.Unimplemented

Unimplemented firmware feature. Fields:

MESSAGE string
Short description of the missing feature.

ArchMsg.Info.LSID0.Firmware.UnknownMessageReceived

Received unknown message.

ArchMsg.Info.LSID0.Firmware.WaitingForInterrupt

Firmware is waiting for IRQ. Fields:

LSID unsigned int
The session ID of the firmware

ArchMsg.Info.LSID0.FirmwareV1.InvalidMessageReceived

Firmware received an invalid meesage from driver. Fields:

MESSAGE enum
The message code.
REASON string
Why the message is invalid.

ArchMsg.Info.LSID0.FirmwareV1.MessageReceived

Firmware received meesage from driver. Fields:

CODE enum
The message code.

ArchMsg.Info.LSID0.FirmwareV1.MessageSent

Firmware sent meesage to driver. Fields:

CODE enum
The message code.

ArchMsg.Info.LSID0.FirmwareV2.InvalidMessageReceived

Firmware received an invalid meesage from driver. Fields:

MESSAGE enum
The message code.
REASON string
Why the message is invalid.

ArchMsg.Info.LSID0.FirmwareV2.MessageReceived

Firmware received meesage from driver. Fields:

CODE enum
The message code.

ArchMsg.Info.LSID0.FirmwareV2.MessageSent

Firmware sent meesage to driver. Fields:

CODE enum
The message code.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Info.LSID1.Firmware.CommandFailure

Command failed. Fields:

CMD unsigned int
: The OMX Command that failed.

ERR unsigned int
: The reported error code.

ArchMsg.Info.LSID1.Firmware.InputBufferReceived

Received input buffer.

ArchMsg.Info.LSID1.Firmware.InvalidBufferHeader

Received invalid buffer header, cannot convert.

ArchMsg.Info.LSID1.Firmware.InvalidCallback

Invalid callback. Fields:

REASON string
: Reason for the invalidity of the callback.

ArchMsg.Info.LSID1.Firmware.OutputBufferReceived

Received output buffer.

ArchMsg.Info.LSID1.Firmware.UnexpectedCallback

Unexpected callback. Fields:

REASON string
: Reason for the invalidity of the callback.

ArchMsg.Info.LSID1.Firmware.Unimplemented

Unimplemented firmware feature. Fields:

MESSAGE string
: Short description of the missing feature.

ArchMsg.Info.LSID1.Firmware.UnknownMessageReceived

Received unknown message.

ArchMsg.Info.LSID1.Firmware.WaitingForInterrupt

Firmware is waiting for IRQ. Fields:

LSID unsigned int
: The session ID of the firmware

ArchMsg.Info.LSID1.FirmwareV1.InvalidMessageReceived

Firmware received an invalid meesage from driver. Fields:

MESSAGE enum
: The message code.

REASON string
: Why the message is invalid.

ArchMsg.Info.LSID1.FirmwareV1.MessageReceived

Firmware received meesage from driver. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CODE enum
: The message code.

ArchMsg.Info.LSID1.FirmwareV1.MessageSent

Firmware sent meesage to driver. Fields:

CODE enum
: The message code.

ArchMsg.Info.LSID1.FirmwareV2.InvalidMessageReceived

Firmware received an invalid meesage from driver. Fields:

MESSAGE enum
: The message code.

REASON string
: Why the message is invalid.

ArchMsg.Info.LSID1.FirmwareV2.MessageReceived

Firmware received meesage from driver. Fields:

CODE enum
: The message code.

ArchMsg.Info.LSID1.FirmwareV2.MessageSent

Firmware sent meesage to driver. Fields:

CODE enum
: The message code.

ArchMsg.Info.LSID2.Firmware.CommandFailure

Command failed. Fields:

CMD unsigned int
: The OMX Command that failed.

ERR unsigned int
: The reported error code.

ArchMsg.Info.LSID2.Firmware.InputBufferReceived

Received input buffer.

ArchMsg.Info.LSID2.Firmware.InvalidBufferHeader

Received invalid buffer header, cannot convert.

ArchMsg.Info.LSID2.Firmware.InvalidCallback

Invalid callback. Fields:

REASON string
: Reason for the invalidity of the callback.

ArchMsg.Info.LSID2.Firmware.OutputBufferReceived

Received output buffer.

ArchMsg.Info.LSID2.Firmware.UnexpectedCallback

Unexpected callback. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

REASON string
Reason for the invalidity of the callback.

ArchMsg.Info.LSID2.Firmware.Unimplemented

Unimplemented firmware feature. Fields:

MESSAGE string
Short description of the missing feature.

ArchMsg.Info.LSID2.Firmware.UnknownMessageReceived

Received unknown message.

ArchMsg.Info.LSID2.Firmware.WaitingForInterrupt

Firmware is waiting for IRQ. Fields:

LSID unsigned int
The session ID of the firmware

ArchMsg.Info.LSID2.FirmwareV1.InvalidMessageReceived

Firmware received an invalid meesage from driver. Fields:

MESSAGE enum
The message code.
REASON string
Why the message is invalid.

ArchMsg.Info.LSID2.FirmwareV1.MessageReceived

Firmware received meesage from driver. Fields:

CODE enum
The message code.

ArchMsg.Info.LSID2.FirmwareV1.MessageSent

Firmware sent meesage to driver. Fields:

CODE enum
The message code.

ArchMsg.Info.LSID2.FirmwareV2.InvalidMessageReceived

Firmware received an invalid meesage from driver. Fields:

MESSAGE enum
The message code.
REASON string
Why the message is invalid.

ArchMsg.Info.LSID2.FirmwareV2.MessageReceived

Firmware received meesage from driver. Fields:

CODE enum
The message code.

ArchMsg.Info.LSID2.FirmwareV2.MessageSent

Firmware sent meesage to driver. Fields:

CODE enum
The message code.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Info.LSID3.Firmware.CommandFailure

Command failed. Fields:

CMD unsigned int
: The OMX Command that failed.

ERR unsigned int
: The reported error code.

ArchMsg.Info.LSID3.Firmware.InputBufferReceived

Received input buffer.

ArchMsg.Info.LSID3.Firmware.InvalidBufferHeader

Received invalid buffer header, cannot convert.

ArchMsg.Info.LSID3.Firmware.InvalidCallback

Invalid callback. Fields:

REASON string
: Reason for the invalidity of the callback.

ArchMsg.Info.LSID3.Firmware.OutputBufferReceived

Received output buffer.

ArchMsg.Info.LSID3.Firmware.UnexpectedCallback

Unexpected callback. Fields:

REASON string
: Reason for the invalidity of the callback.

ArchMsg.Info.LSID3.Firmware.Unimplemented

Unimplemented firmware feature. Fields:

MESSAGE string
: Short description of the missing feature.

ArchMsg.Info.LSID3.Firmware.UnknownMessageReceived

Received unknown message.

ArchMsg.Info.LSID3.Firmware.WaitingForInterrupt

Firmware is waiting for IRQ. Fields:

LSID unsigned int
: The session ID of the firmware

ArchMsg.Info.LSID3.FirmwareV1.InvalidMessageReceived

Firmware received an invalid meesage from driver. Fields:

MESSAGE enum
: The message code.

REASON string
: Why the message is invalid.

ArchMsg.Info.LSID3.FirmwareV1.MessageReceived

Firmware received meesage from driver. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CODE enum
: The message code.

ArchMsg.Info.LSID3.FirmwareV1.MessageSent

Firmware sent meesage to driver. Fields:

CODE enum
: The message code.

ArchMsg.Info.LSID3.FirmwareV2.InvalidMessageReceived

Firmware received an invalid meesage from driver. Fields:

MESSAGE enum
: The message code.

REASON string
: Why the message is invalid.

ArchMsg.Info.LSID3.FirmwareV2.MessageReceived

Firmware received meesage from driver. Fields:

CODE enum
: The message code.

ArchMsg.Info.LSID3.FirmwareV2.MessageSent

Firmware sent meesage to driver. Fields:

CODE enum
: The message code.

ArchMsg.Info.MmuTranslateAccessFailure

MMU translation failed (mismatched access rights). Fields:

ACC enum
: Page access rights

INTENT enum
: Intent of the memory transaction, e.g. read+execute, read-only, write

VA unsigned int
: MVE virtual address to be translated

ArchMsg.Info.MmuTranslateLxFailure

MMU translation failed (entry missing). Fields:

LEVEL unsigned int
: The translation level at which the failure occurred; either 1 or 2

VA unsigned int
: MVE virtual address to be translated

ArchMsg.Info.MmuTranslateSuccess

MMU translation succeeded. Fields:

ACC enum
: Access rights to the address/page

ATTR enum
: Bus attributes for the memory transactions

PA unsigned int
: Resultant physical address

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VA unsigned int
: MVE virtual address to be translated

ArchMsg.Info.NSReadFromSecureRam

NS read of secure RAM attempted. Fields:

OFFSET unsigned int
: Access offset

ArchMsg.Info.NSReadFromSecureRegister

NS read of secure register attempted. Fields:

NAME string
: Register name

OFFSET unsigned int
: Access offset

ArchMsg.Info.NSWriteToSecureRam

NS write to RAM of secure session attempted. Fields:

OFFSET unsigned int
: Access offset

ArchMsg.Info.NSWriteToSecureRegister

NS write to secure register attempted. Fields:

DATA unsigned int
: The data written to the address

NAME string
: Register name

OFFSET unsigned int
: Access offset

ArchMsg.Info.ReadFromWriteOnlyRegister

Read of write-only register attempted. Fields:

NAME string
: Register name

NSECURE bool
: True when the access is not a secure access

OFFSET unsigned int
: Access offset

ArchMsg.Info.ReadInReset

Attempt to read register during reset. Fields:

NSECURE bool
: True when the access is not a secure access

OFFSET unsigned int
: Access offset

ArchMsg.Info.RegisterRead

Register has been read. Fields:

DATA unsigned int
: The data returned by the component

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NAME string
: Register name

NSECURE bool
: True when the access is not a secure access

OFFSET unsigned int
: Register access offset

ArchMsg.Info.RegisterWrite

Register written. Fields:

DATA unsigned int
: The data written to the register

NAME string
: Register name

NSECURE bool
: True when the access is not a secure access

OFFSET unsigned int
: Register access offset

ArchMsg.Info.SessionStarted

Session started. Fields:

COREMASK unsigned int
: 8-bit 1-hot encoded value of newly-assigned cores

LSID unsigned int
: Session ID

NUMCORES unsigned int
: Number of cores for the session

ArchMsg.Info.SessionStopped

Session stopped. Fields:

COREMASK unsigned int
: 8-bit 1-hot encoded value of freed cores

LSID unsigned int
: Session ID

NUMCORES unsigned int
: Number of cores for the session

ArchMsg.Info.UnalignedRead

Unaligned read attempted. Fields:

NSECURE bool
: True when the access is not a secure access

OFFSET unsigned int
: Access offset

ArchMsg.Info.UnalignedWrite

Unaligned write attempted. Fields:

DATA unsigned int
: The data written to the address

NSECURE bool
: True when the access is not a secure access

OFFSET unsigned int
: Access offset

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Info.WriteInReset

Attempt to write register during reset. Fields:

DATA unsigned int
: The data written to the address

NSECURE bool
: True when the access is not a secure access

OFFSET unsigned int
: Access offset

ArchMsg.Info.WriteToReadOnlyRegister

Write to read-only register attempted. Fields:

DATA unsigned int
: The data written to the address

NAME string
: Register name

NSECURE bool
: True when the access is not a secure access

OFFSET unsigned int
: Access offset

ArchMsg.Info.WriteToSecureRegisterOfUnallocatedLSID

Write to secure register of unallocated LSID attempted. Fields:

DATA unsigned int
: The data written to the address

NAME string
: Register name

OFFSET unsigned int
: Access offset

ArchMsg.Warning.AxiReadFailure

AXI read failed. Fields:

ADDR unsigned int
: The address of the failed transaction

PURPOSE string
: The reason for the AXI transaction

WIDTH unsigned int
: The access width of the transaction

ArchMsg.Warning.AxiWriteFailure

AXI write failed. Fields:

ADDR unsigned int
: The address of the failed transaction

PURPOSE string
: The reason for the AXI transaction

WIDTH unsigned int
: The access width of the transaction

ArchMsg.Warning.InsecureNSAIDConfiguration

Potentially insecure NSAID configuration. Fields:

NSAID string
: The NSAID whose value is potentially causing insecure configurations.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.LSID0.Firmware.BackendError

Backend reported error when executing command. Fields:

COMMAND string
: String describing the command.

ArchMsg.Warning.LSID0.Firmware.ComponentInstantiationFailure

Failed to instantiate OMX component. Fields:

ERR unsigned int
: The error returned by the OMX core.

NAME string
: The name or role of the OMX IL component.

ArchMsg.Warning.LSID0.Firmware.ComponentNotInitialized

OMX component failed to initialize, refusing to use it.

ArchMsg.Warning.LSID0.Firmware.InvalidFirmwareLoaded

Driver loaded an invalid firmware image.

ArchMsg.Warning.LSID0.Firmware.UnexpectedSessionLoaded

Loaded an unexpected session from the firmware's memory map. Creating a blank session.

ArchMsg.Warning.LSID0.Firmware.UnsupportedFwProtocolVersion

Encountered unsupported protocol version. Fields:

MAJOR unsigned int
: Major version number of the firmware protocol.

MINOR unsigned int
: Minor version number of the firmware protocol.

ArchMsg.Warning.LSID0.FirmwareV2.UnableToChangeQpMidStream

Changing Qp mid-stream is not supported by the model.

ArchMsg.Warning.LSID1.Firmware.BackendError

Backend reported error when executing command. Fields:

COMMAND string
: String describing the command.

ArchMsg.Warning.LSID1.Firmware.ComponentInstantiationFailure

Failed to instantiate OMX component. Fields:

ERR unsigned int
: The error returned by the OMX core.

NAME string
: The name or role of the OMX IL component.

ArchMsg.Warning.LSID1.Firmware.ComponentNotInitialized

OMX component failed to initialize, refusing to use it.

ArchMsg.Warning.LSID1.Firmware.InvalidFirmwareLoaded

Driver loaded an invalid firmware image.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.LSID1.Firmware.UnexpectedSessionLoaded

Loaded an unexpected session from the firmware's memory map. Creating a blank session.

ArchMsg.Warning.LSID1.Firmware.UnsupportedFwProtocolVersion

Encountered unsupported protocol version. Fields:

MAJOR unsigned int
: Major version number of the firmware protocol.

MINOR unsigned int
: Minor version number of the firmware protocol.

ArchMsg.Warning.LSID1.FirmwareV2.UnableToChangeQpMidStream

Changing Qp mid-stream is not supported by the model.

ArchMsg.Warning.LSID2.Firmware.BackendError

Backend reported error when executing command. Fields:

COMMAND string
: String describing the command.

ArchMsg.Warning.LSID2.Firmware.ComponentInstantiationFailure

Failed to instantiate OMX component. Fields:

ERR unsigned int
: The error returned by the OMX core.

NAME string
: The name or role of the OMX IL component.

ArchMsg.Warning.LSID2.Firmware.ComponentNotInitialized

OMX component failed to initialize, refusing to use it.

ArchMsg.Warning.LSID2.Firmware.InvalidFirmwareLoaded

Driver loaded an invalid firmware image.

ArchMsg.Warning.LSID2.Firmware.UnexpectedSessionLoaded

Loaded an unexpected session from the firmware's memory map. Creating a blank session.

ArchMsg.Warning.LSID2.Firmware.UnsupportedFwProtocolVersion

Encountered unsupported protocol version. Fields:

MAJOR unsigned int
: Major version number of the firmware protocol.

MINOR unsigned int
: Minor version number of the firmware protocol.

ArchMsg.Warning.LSID2.FirmwareV2.UnableToChangeQpMidStream

Changing Qp mid-stream is not supported by the model.

ArchMsg.Warning.LSID3.Firmware.BackendError

Backend reported error when executing command. Fields:

COMMAND string
: String describing the command.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.LSID3.Firmware.ComponentInstantiationFailure

Failed to instantiate OMX component. Fields:

ERR unsigned int
: The error returned by the OMX core.

NAME string
: The name or role of the OMX IL component.

ArchMsg.Warning.LSID3.Firmware.ComponentNotInitialized

OMX component failed to initialize, refusing to use it.

ArchMsg.Warning.LSID3.Firmware.InvalidFirmwareLoaded

Driver loaded an invalid firmware image.

ArchMsg.Warning.LSID3.Firmware.UnexpectedSessionLoaded

Loaded an unexpected session from the firmware's memory map. Creating a blank session.

ArchMsg.Warning.LSID3.Firmware.UnsupportedFwProtocolVersion

Encountered unsupported protocol version. Fields:

MAJOR unsigned int
: Major version number of the firmware protocol.

MINOR unsigned int
: Minor version number of the firmware protocol.

ArchMsg.Warning.LSID3.FirmwareV2.UnableToChangeQpMidStream

Changing Qp mid-stream is not supported by the model.

ArchMsg.Warning.ReadFromRamOfUnallocatedSession

Attempt to read from unallocated session. Fields:

OFFSET unsigned int
: Access offset

ArchMsg.Warning.ReservedProtctrlConfig

Mismatch between CPS field and L1R/L2R fields in PROTCTRL. Accepted combinations are: CPS == 0 && L1R == L2R == 0; CPS == 1 && L1R == L2R == 3. Using any other combination is reserved and undefined. .

ArchMsg.Warning.UninitialisedHardware

Attempt to write to register before hardware is initialised. Fields:

DATA unsigned int
: Data written to register

REGNAME string
: The name of the register

ArchMsg.Warning.WriteOfInvalidDataToRegister

Write of invalid data to register attempted. Fields:

DATA unsigned int
: The data written to the address

NAME string
: Register name

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

REASON enum
A short description of why the value was rejected

ArchMsg.Warning.WriteToRamOfUnallocatedSession

Write to session RAM of unallocated session attempted. Fields:

OFFSET unsigned int
Access offset

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.116 V61 - trace

This section describes the trace sources.

ArchMsg.Info.IgnoredRead

Read to reserved area attempted. Fields:

NSECURE bool
: True when the access is not a secure access

OFFSET unsigned int
: Access offset

ArchMsg.Info.IgnoredWrite

Write to reserved area attempted. Fields:

DATA unsigned int
: The data written to the address

NSECURE bool
: True when the access is not a secure access

OFFSET unsigned int
: Access offset

ArchMsg.Info.JobqueueUpdated

Job queue updated. Fields:

NEWVAL unsigned int
: New JOBQUEUE value

OLDVAL unsigned int
: Previous JOBQUEUE value

ArchMsg.Info.LSID0.Firmware.CommandFailure

Command failed. Fields:

CMD unsigned int
: The OMX Command that failed.

ERR unsigned int
: The reported error code.

ArchMsg.Info.LSID0.Firmware.InputBufferReceived

Received input buffer.

ArchMsg.Info.LSID0.Firmware.InvalidBufferHeader

Received invalid buffer header, cannot convert.

ArchMsg.Info.LSID0.Firmware.InvalidCallback

Invalid callback. Fields:

REASON string
: Reason for the invalidity of the callback.

ArchMsg.Info.LSID0.Firmware.OutputBufferReceived

Received output buffer.

ArchMsg.Info.LSID0.Firmware.UnexpectedCallback

Unexpected callback. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

REASON string
Reason for the invalidity of the callback.

ArchMsg.Info.LSID0.Firmware.Unimplemented

Unimplemented firmware feature. Fields:

MESSAGE string
Short description of the missing feature.

ArchMsg.Info.LSID0.Firmware.UnknownMessageReceived

Received unknown message.

ArchMsg.Info.LSID0.Firmware.WaitingForInterrupt

Firmware is waiting for IRQ. Fields:

LSID unsigned int
The session ID of the firmware

ArchMsg.Info.LSID0.FirmwareV1.InvalidMessageReceived

Firmware received an invalid meesage from driver. Fields:

MESSAGE enum
The message code.
REASON string
Why the message is invalid.

ArchMsg.Info.LSID0.FirmwareV1.MessageReceived

Firmware received meesage from driver. Fields:

CODE enum
The message code.

ArchMsg.Info.LSID0.FirmwareV1.MessageSent

Firmware sent meesage to driver. Fields:

CODE enum
The message code.

ArchMsg.Info.LSID0.FirmwareV2.InvalidMessageReceived

Firmware received an invalid meesage from driver. Fields:

MESSAGE enum
The message code.
REASON string
Why the message is invalid.

ArchMsg.Info.LSID0.FirmwareV2.MessageReceived

Firmware received meesage from driver. Fields:

CODE enum
The message code.

ArchMsg.Info.LSID0.FirmwareV2.MessageSent

Firmware sent meesage to driver. Fields:

CODE enum
The message code.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Info.LSID1.Firmware.CommandFailure

Command failed. Fields:

CMD unsigned int
: The OMX Command that failed.

ERR unsigned int
: The reported error code.

ArchMsg.Info.LSID1.Firmware.InputBufferReceived

Received input buffer.

ArchMsg.Info.LSID1.Firmware.InvalidBufferHeader

Received invalid buffer header, cannot convert.

ArchMsg.Info.LSID1.Firmware.InvalidCallback

Invalid callback. Fields:

REASON string
: Reason for the invalidity of the callback.

ArchMsg.Info.LSID1.Firmware.OutputBufferReceived

Received output buffer.

ArchMsg.Info.LSID1.Firmware.UnexpectedCallback

Unexpected callback. Fields:

REASON string
: Reason for the invalidity of the callback.

ArchMsg.Info.LSID1.Firmware.Unimplemented

Unimplemented firmware feature. Fields:

MESSAGE string
: Short description of the missing feature.

ArchMsg.Info.LSID1.Firmware.UnknownMessageReceived

Received unknown message.

ArchMsg.Info.LSID1.Firmware.WaitingForInterrupt

Firmware is waiting for IRQ. Fields:

LSID unsigned int
: The session ID of the firmware

ArchMsg.Info.LSID1.FirmwareV1.InvalidMessageReceived

Firmware received an invalid meesage from driver. Fields:

MESSAGE enum
: The message code.

REASON string
: Why the message is invalid.

ArchMsg.Info.LSID1.FirmwareV1.MessageReceived

Firmware received meesage from driver. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CODE enum
: The message code.

ArchMsg.Info.LSID1.FirmwareV1.MessageSent

Firmware sent meesage to driver. Fields:

CODE enum
: The message code.

ArchMsg.Info.LSID1.FirmwareV2.InvalidMessageReceived

Firmware received an invalid meesage from driver. Fields:

MESSAGE enum
: The message code.

REASON string
: Why the message is invalid.

ArchMsg.Info.LSID1.FirmwareV2.MessageReceived

Firmware received meesage from driver. Fields:

CODE enum
: The message code.

ArchMsg.Info.LSID1.FirmwareV2.MessageSent

Firmware sent meesage to driver. Fields:

CODE enum
: The message code.

ArchMsg.Info.LSID2.Firmware.CommandFailure

Command failed. Fields:

CMD unsigned int
: The OMX Command that failed.

ERR unsigned int
: The reported error code.

ArchMsg.Info.LSID2.Firmware.InputBufferReceived

Received input buffer.

ArchMsg.Info.LSID2.Firmware.InvalidBufferHeader

Received invalid buffer header, cannot convert.

ArchMsg.Info.LSID2.Firmware.InvalidCallback

Invalid callback. Fields:

REASON string
: Reason for the invalidity of the callback.

ArchMsg.Info.LSID2.Firmware.OutputBufferReceived

Received output buffer.

ArchMsg.Info.LSID2.Firmware.UnexpectedCallback

Unexpected callback. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

REASON string
Reason for the invalidity of the callback.

ArchMsg.Info.LSID2.Firmware.Unimplemented

Unimplemented firmware feature. Fields:

MESSAGE string
Short description of the missing feature.

ArchMsg.Info.LSID2.Firmware.UnknownMessageReceived

Received unknown message.

ArchMsg.Info.LSID2.Firmware.WaitingForInterrupt

Firmware is waiting for IRQ. Fields:

LSID unsigned int
The session ID of the firmware

ArchMsg.Info.LSID2.FirmwareV1.InvalidMessageReceived

Firmware received an invalid meesage from driver. Fields:

MESSAGE enum
The message code.
REASON string
Why the message is invalid.

ArchMsg.Info.LSID2.FirmwareV1.MessageReceived

Firmware received meesage from driver. Fields:

CODE enum
The message code.

ArchMsg.Info.LSID2.FirmwareV1.MessageSent

Firmware sent meesage to driver. Fields:

CODE enum
The message code.

ArchMsg.Info.LSID2.FirmwareV2.InvalidMessageReceived

Firmware received an invalid meesage from driver. Fields:

MESSAGE enum
The message code.
REASON string
Why the message is invalid.

ArchMsg.Info.LSID2.FirmwareV2.MessageReceived

Firmware received meesage from driver. Fields:

CODE enum
The message code.

ArchMsg.Info.LSID2.FirmwareV2.MessageSent

Firmware sent meesage to driver. Fields:

CODE enum
The message code.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Info.LSID3.Firmware.CommandFailure

Command failed. Fields:

CMD unsigned int
: The OMX Command that failed.

ERR unsigned int
: The reported error code.

ArchMsg.Info.LSID3.Firmware.InputBufferReceived

Received input buffer.

ArchMsg.Info.LSID3.Firmware.InvalidBufferHeader

Received invalid buffer header, cannot convert.

ArchMsg.Info.LSID3.Firmware.InvalidCallback

Invalid callback. Fields:

REASON string
: Reason for the invalidity of the callback.

ArchMsg.Info.LSID3.Firmware.OutputBufferReceived

Received output buffer.

ArchMsg.Info.LSID3.Firmware.UnexpectedCallback

Unexpected callback. Fields:

REASON string
: Reason for the invalidity of the callback.

ArchMsg.Info.LSID3.Firmware.Unimplemented

Unimplemented firmware feature. Fields:

MESSAGE string
: Short description of the missing feature.

ArchMsg.Info.LSID3.Firmware.UnknownMessageReceived

Received unknown message.

ArchMsg.Info.LSID3.Firmware.WaitingForInterrupt

Firmware is waiting for IRQ. Fields:

LSID unsigned int
: The session ID of the firmware

ArchMsg.Info.LSID3.FirmwareV1.InvalidMessageReceived

Firmware received an invalid meesage from driver. Fields:

MESSAGE enum
: The message code.

REASON string
: Why the message is invalid.

ArchMsg.Info.LSID3.FirmwareV1.MessageReceived

Firmware received meesage from driver. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CODE enum
: The message code.

ArchMsg.Info.LSID3.FirmwareV1.MessageSent

Firmware sent meesage to driver. Fields:

CODE enum
: The message code.

ArchMsg.Info.LSID3.FirmwareV2.InvalidMessageReceived

Firmware received an invalid meesage from driver. Fields:

MESSAGE enum
: The message code.

REASON string
: Why the message is invalid.

ArchMsg.Info.LSID3.FirmwareV2.MessageReceived

Firmware received meesage from driver. Fields:

CODE enum
: The message code.

ArchMsg.Info.LSID3.FirmwareV2.MessageSent

Firmware sent meesage to driver. Fields:

CODE enum
: The message code.

ArchMsg.Info.MmuTranslateAccessFailure

MMU translation failed (mismatched access rights). Fields:

ACC enum
: Page access rights

INTENT enum
: Intent of the memory transaction, e.g. read+execute, read-only, write

VA unsigned int
: MVE virtual address to be translated

ArchMsg.Info.MmuTranslateLxFailure

MMU translation failed (entry missing). Fields:

LEVEL unsigned int
: The translation level at which the failure occurred; either 1 or 2

VA unsigned int
: MVE virtual address to be translated

ArchMsg.Info.MmuTranslateSuccess

MMU translation succeeded. Fields:

ACC enum
: Access rights to the address/page

ATTR enum
: Bus attributes for the memory transactions

PA unsigned int
: Resultant physical address

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

VA unsigned int
: MVE virtual address to be translated

ArchMsg.Info.NSReadFromSecureRam

NS read of secure RAM attempted. Fields:

OFFSET unsigned int
: Access offset

ArchMsg.Info.NSReadFromSecureRegister

NS read of secure register attempted. Fields:

NAME string
: Register name

OFFSET unsigned int
: Access offset

ArchMsg.Info.NSWriteToSecureRam

NS write to RAM of secure session attempted. Fields:

OFFSET unsigned int
: Access offset

ArchMsg.Info.NSWriteToSecureRegister

NS write to secure register attempted. Fields:

DATA unsigned int
: The data written to the address

NAME string
: Register name

OFFSET unsigned int
: Access offset

ArchMsg.Info.ReadFromWriteOnlyRegister

Read of write-only register attempted. Fields:

NAME string
: Register name

NSECURE bool
: True when the access is not a secure access

OFFSET unsigned int
: Access offset

ArchMsg.Info.ReadInReset

Attempt to read register during reset. Fields:

NSECURE bool
: True when the access is not a secure access

OFFSET unsigned int
: Access offset

ArchMsg.Info.RegisterRead

Register has been read. Fields:

DATA unsigned int
: The data returned by the component

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

NAME string
: Register name

NSECURE bool
: True when the access is not a secure access

OFFSET unsigned int
: Register access offset

ArchMsg.Info.RegisterWrite

Register written. Fields:

DATA unsigned int
: The data written to the register

NAME string
: Register name

NSECURE bool
: True when the access is not a secure access

OFFSET unsigned int
: Register access offset

ArchMsg.Info.SessionStarted

Session started. Fields:

COREMASK unsigned int
: 8-bit 1-hot encoded value of newly-assigned cores

LSID unsigned int
: Session ID

NUMCORES unsigned int
: Number of cores for the session

ArchMsg.Info.SessionStopped

Session stopped. Fields:

COREMASK unsigned int
: 8-bit 1-hot encoded value of freed cores

LSID unsigned int
: Session ID

NUMCORES unsigned int
: Number of cores for the session

ArchMsg.Info.UnalignedRead

Unaligned read attempted. Fields:

NSECURE bool
: True when the access is not a secure access

OFFSET unsigned int
: Access offset

ArchMsg.Info.UnalignedWrite

Unaligned write attempted. Fields:

DATA unsigned int
: The data written to the address

NSECURE bool
: True when the access is not a secure access

OFFSET unsigned int
: Access offset

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Info.WriteInReset

Attempt to write register during reset. Fields:

DATA unsigned int
: The data written to the address

NSECURE bool
: True when the access is not a secure access

OFFSET unsigned int
: Access offset

ArchMsg.Info.WriteToReadOnlyRegister

Write to read-only register attempted. Fields:

DATA unsigned int
: The data written to the address

NAME string
: Register name

NSECURE bool
: True when the access is not a secure access

OFFSET unsigned int
: Access offset

ArchMsg.Info.WriteToSecureRegisterOfUnallocatedLSID

Write to secure register of unallocated LSID attempted. Fields:

DATA unsigned int
: The data written to the address

NAME string
: Register name

OFFSET unsigned int
: Access offset

ArchMsg.Warning.AxiReadFailure

AXI read failed. Fields:

ADDR unsigned int
: The address of the failed transaction

PURPOSE string
: The reason for the AXI transaction

WIDTH unsigned int
: The access width of the transaction

ArchMsg.Warning.AxiWriteFailure

AXI write failed. Fields:

ADDR unsigned int
: The address of the failed transaction

PURPOSE string
: The reason for the AXI transaction

WIDTH unsigned int
: The access width of the transaction

ArchMsg.Warning.InsecureNSAIDConfiguration

Potentially insecure NSAID configuration. Fields:

NSAID string
: The NSAID whose value is potentially causing insecure configurations.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.LSID0.Firmware.BackendError

Backend reported error when executing command. Fields:

COMMAND string
: String describing the command.

ArchMsg.Warning.LSID0.Firmware.ComponentInstantiationFailure

Failed to instantiate OMX component. Fields:

ERR unsigned int
: The error returned by the OMX core.

NAME string
: The name or role of the OMX IL component.

ArchMsg.Warning.LSID0.Firmware.ComponentNotInitialized

OMX component failed to initialize, refusing to use it.

ArchMsg.Warning.LSID0.Firmware.InvalidFirmwareLoaded

Driver loaded an invalid firmware image.

ArchMsg.Warning.LSID0.Firmware.UnexpectedSessionLoaded

Loaded an unexpected session from the firmware's memory map. Creating a blank session.

ArchMsg.Warning.LSID0.Firmware.UnsupportedFwProtocolVersion

Encountered unsupported protocol version. Fields:

MAJOR unsigned int
: Major version number of the firmware protocol.

MINOR unsigned int
: Minor version number of the firmware protocol.

ArchMsg.Warning.LSID0.FirmwareV2.UnableToChangeQpMidStream

Changing Qp mid-stream is not supported by the model.

ArchMsg.Warning.LSID1.Firmware.BackendError

Backend reported error when executing command. Fields:

COMMAND string
: String describing the command.

ArchMsg.Warning.LSID1.Firmware.ComponentInstantiationFailure

Failed to instantiate OMX component. Fields:

ERR unsigned int
: The error returned by the OMX core.

NAME string
: The name or role of the OMX IL component.

ArchMsg.Warning.LSID1.Firmware.ComponentNotInitialized

OMX component failed to initialize, refusing to use it.

ArchMsg.Warning.LSID1.Firmware.InvalidFirmwareLoaded

Driver loaded an invalid firmware image.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.LSID1.Firmware.UnexpectedSessionLoaded

Loaded an unexpected session from the firmware's memory map. Creating a blank session.

ArchMsg.Warning.LSID1.Firmware.UnsupportedFwProtocolVersion

Encountered unsupported protocol version. Fields:

MAJOR unsigned int
: Major version number of the firmware protocol.

MINOR unsigned int
: Minor version number of the firmware protocol.

ArchMsg.Warning.LSID1.FirmwareV2.UnableToChangeQpMidStream

Changing Qp mid-stream is not supported by the model.

ArchMsg.Warning.LSID2.Firmware.BackendError

Backend reported error when executing command. Fields:

COMMAND string
: String describing the command.

ArchMsg.Warning.LSID2.Firmware.ComponentInstantiationFailure

Failed to instantiate OMX component. Fields:

ERR unsigned int
: The error returned by the OMX core.

NAME string
: The name or role of the OMX IL component.

ArchMsg.Warning.LSID2.Firmware.ComponentNotInitialized

OMX component failed to initialize, refusing to use it.

ArchMsg.Warning.LSID2.Firmware.InvalidFirmwareLoaded

Driver loaded an invalid firmware image.

ArchMsg.Warning.LSID2.Firmware.UnexpectedSessionLoaded

Loaded an unexpected session from the firmware's memory map. Creating a blank session.

ArchMsg.Warning.LSID2.Firmware.UnsupportedFwProtocolVersion

Encountered unsupported protocol version. Fields:

MAJOR unsigned int
: Major version number of the firmware protocol.

MINOR unsigned int
: Minor version number of the firmware protocol.

ArchMsg.Warning.LSID2.FirmwareV2.UnableToChangeQpMidStream

Changing Qp mid-stream is not supported by the model.

ArchMsg.Warning.LSID3.Firmware.BackendError

Backend reported error when executing command. Fields:

COMMAND string
: String describing the command.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.LSID3.Firmware.ComponentInstantiationFailure

Failed to instantiate OMX component. Fields:

ERR unsigned int
: The error returned by the OMX core.

NAME string
: The name or role of the OMX IL component.

ArchMsg.Warning.LSID3.Firmware.ComponentNotInitialized

OMX component failed to initialize, refusing to use it.

ArchMsg.Warning.LSID3.Firmware.InvalidFirmwareLoaded

Driver loaded an invalid firmware image.

ArchMsg.Warning.LSID3.Firmware.UnexpectedSessionLoaded

Loaded an unexpected session from the firmware's memory map. Creating a blank session.

ArchMsg.Warning.LSID3.Firmware.UnsupportedFwProtocolVersion

Encountered unsupported protocol version. Fields:

MAJOR unsigned int
: Major version number of the firmware protocol.

MINOR unsigned int
: Minor version number of the firmware protocol.

ArchMsg.Warning.LSID3.FirmwareV2.UnableToChangeQpMidStream

Changing Qp mid-stream is not supported by the model.

ArchMsg.Warning.ReadFromRamOfUnallocatedSession

Attempt to read from unallocated session. Fields:

OFFSET unsigned int
: Access offset

ArchMsg.Warning.ReservedProtctrlConfig

Mismatch between CPS field and L1R/L2R fields in PROTCTRL. Accepted combinations are: CPS == 0 && L1R == L2R == 0; CPS == 1 && L1R == L2R == 3. Using any other combination is reserved and undefined. .

ArchMsg.Warning.UninitialisedHardware

Attempt to write to register before hardware is initialised. Fields:

DATA unsigned int
: Data written to register

REGNAME string
: The name of the register

ArchMsg.Warning.WriteOfInvalidDataToRegister

Write of invalid data to register attempted. Fields:

DATA unsigned int
: The data written to the address

NAME string
: Register name

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

REASON enum
A short description of why the value was rejected

ArchMsg.Warning.WriteToRamOfUnallocatedSession

Write to session RAM of unallocated session attempted. Fields:

OFFSET unsigned int
Access offset

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.117 Value64Logger - trace

This section describes the trace sources.

VALUE

Input signal to ValueLogger driven. Fields:

Value `unsigned int`
: Value of signal being driven

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.118 ValueLogger - trace

This section describes the trace sources.

VALUE

Input signal to ValueLogger driven. Fields:

Value `unsigned int`
Value of signal being driven

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.119 VirtioBlockDevice - trace

This section describes the trace sources.

CHECKPOINT_MESSAGE

Report error messages from the checkpointing process. Fields:

message string
: Message contents.

CHECKPOINT_RESTORE_END

Checkpoint restore completed.

CHECKPOINT_RESTORE_START

Checkpoint restore about to start.

CHECKPOINT_SAVE_END

Checkpoint save completed.

CHECKPOINT_SAVE_START

Checkpoint save about to start.

data_read

The data read by the device. Fields:

addr unsigned int
: Address.

data unsigned int
: The data read.

ns bool
: Non-secure access.

data_read_abort

The data read by the device aborted. Fields:

addr unsigned int
: Address.

ns bool
: Non-secure access.

size_in_bytes unsigned int
: The data size in bytes.

data_read_start

The data read by the device. Fields:

addr unsigned int
: Address.

ns bool
: Non-secure access.

size_in_bytes unsigned int
: The size to read.

data_write

The data write by the device completed successfully. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

addr unsigned int
: Address.

ns bool
: Non-secure access.

size_in_bytes unsigned int
: The size of the write that completed successfully.

data_write_abort

The data write by the device aborted. Fields:

addr unsigned int
: Address.

ns bool
: Non-secure access.

size_in_bytes unsigned int
: The size of the write that aborted.

data_write_start

The data attempted to be written by the device. Fields:

addr unsigned int
: Address.

data unsigned int
: The data attempted to be written.

ns bool
: Non-secure access.

register_read

A read of the register file. Fields:

addr unsigned int
: Address.

data unsigned int
: Data read.

what enum
: What region of MMIO space it accessed.

register_read_abort

A read of the register file caused an abort. Fields:

addr unsigned int
: Address.

what enum
: What region of MMIO space it accessed.

width_in_bytes unsigned int
: Width of data read.

register_write

A write of the register file. Fields:

addr unsigned int
: Address.

data unsigned int
: Data written.

what enum
: What region of MMIO space it accessed.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

register_write_abort

A write of the register file caused an abort. Fields:

addr `unsigned int`
: Address.

data `unsigned int`
: Data attempted to be written.

what `enum`
: What region of MMIO space it accessed.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.120 VirtioP9Device - trace

This section describes the trace sources.

CHECKPOINT_MESSAGE

Report error messages from the checkpointing process. Fields:

message `string`
: Message contents.

CHECKPOINT_RESTORE_END

Checkpoint restore completed.

CHECKPOINT_RESTORE_START

Checkpoint restore about to start.

CHECKPOINT_SAVE_END

Checkpoint save completed.

CHECKPOINT_SAVE_START

Checkpoint save about to start.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.121 VirtioPCIBlockDevice - trace

This section describes the trace sources.

CHECKPOINT_MESSAGE

Report error messages from the checkpointing process. Fields:

message string
: Message contents.

CHECKPOINT_RESTORE_END

Checkpoint restore completed.

CHECKPOINT_RESTORE_START

Checkpoint restore about to start.

CHECKPOINT_SAVE_END

Checkpoint save completed.

CHECKPOINT_SAVE_START

Checkpoint save about to start.

data_read

The data read by the device. Fields:

addr unsigned int
: Address.

data unsigned int
: The data read.

ns bool
: Non-secure access.

data_read_abort

The data read by the device aborted. Fields:

addr unsigned int
: Address.

ns bool
: Non-secure access.

size_in_bytes unsigned int
: The data size in bytes.

data_read_start

The data read by the device. Fields:

addr unsigned int
: Address.

ns bool
: Non-secure access.

size_in_bytes unsigned int
: The size to read.

data_write

The data write by the device completed successfully. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

addr unsigned int
: Address.

ns bool
: Non-secure access.

size_in_bytes unsigned int
: The size of the write that completed successfully.

data_write_abort

The data write by the device aborted. Fields:

addr unsigned int
: Address.

ns bool
: Non-secure access.

size_in_bytes unsigned int
: The size of the write that aborted.

data_write_start

The data attempted to be written by the device. Fields:

addr unsigned int
: Address.

data unsigned int
: The data attempted to be written.

ns bool
: Non-secure access.

register_read

A read of the register file. Fields:

addr unsigned int
: Address.

data unsigned int
: Data read.

what enum
: What region of MMIO space it accessed.

register_read_abort

A read of the register file caused an abort. Fields:

addr unsigned int
: Address.

what enum
: What region of MMIO space it accessed.

width_in_bytes unsigned int
: Width of data read.

register_write

A write of the register file. Fields:

addr unsigned int
: Address.

data unsigned int
: Data written.

what enum
: What region of MMIO space it accessed.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

register_write_abort

A write of the register file caused an abort. Fields:

addr `unsigned int`
: Address.

data `unsigned int`
: Data attempted to be written.

what `enum`
: What region of MMIO space it accessed.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.122 VirtualEthernetCrossover - trace

This section describes the trace sources.

FRAME

Trace Ethernet frames as they go through. Fields:

data string
: ASCII encoded packet data. Each line starts with a hex packet offset (in bytes) followed by up to 16 space separated bytes encoded as 2 hex digits. This data can be read by the tool text2pcap to generate a file which can be read by wireshark.

source enum
: Identifies the source port from which the frame originated

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.123 atc - trace

This section describes the trace sources.

ArchMsg.Error.error

These messages are about activity occurring on the ATC that is considered an error. Fields:

output string
: The stream output

ArchMsg.Info.info

These are information messages about what is happening in the ATC. Fields:

output string
: The stream output

ArchMsg.Warning.atc_pri_allocation_changed_whilst_not_stopped

The PRI has PRGs in flight that were using credit_used credits, however, the PRI interface was disabled and the allocation of credits changed. Fields:

credits_used unsigned int
: The number of credits currently in use by in-flight PRGs

new_allocation unsigned int
: The new allocation of credits set in the PRI header.

ArchMsg.Warning.atc_pri_allocation_reduced_below_current_credits_used

The PRI has PRGs in flight that were using credit_used credits, however, the PRI interface was disabled and the allocation of credits changed to a value less than that of those that are currently in use. The SW should probably of first have serviced the outstanding PRGs and then waited for the interface to report Stopped before reducing the allocation. The PCIe spec is unclear as to the required behaviour. The model will continue to count credits. It will not allow the PRI interface to be re-enabled until Stopped (this is an implementation defined choice) and so when successfully re-enabled will then obey the new allocation. Fields:

credits_used unsigned int
: The number of credits currently in use by in-flight PRGs.

new_allocation unsigned int
: The new allocation of credits set in the PRI header.

ArchMsg.Warning.atc_pri_members_of_prg_inconsistent

A client has formed a PRG and all the members of the PRG should have the same execute mode and privilege request in. However, this PRG member has different ones to the first. PCIe says that the effect is undefined. The trace source pri_members_of_prg traces all members of the group. Fields:

PRGIndex unsigned int
: The PRGIndex of this request.

expected_execute bool
: The expected execute permission requested.

expected_priv bool
: The expected privilege permission requested.

member_index unsigned int
: The index of this member within the PRG.

number_of_members unsigned int
: The number of members in this PRG.

page_address unsigned int
: Page address of the request.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

priv bool
: Privileged permissions requested. Valid only for those requests with a PASID (SubstreamID).

rwx enum
: Requested Read/Write/Execute permissions. Execute permission are only available for those requests with a PASID (SubstreamID), and read permission will automatically be asked for as well.

streamid unsigned int
: The StreamID of the PRI request.

substreamid unsigned int
: The SubstreamID sent with the PRG request.

trans_id unsigned int
: The transaction id of the failing ATS transaction.

ArchMsg.Warning.atc_pri_reset_received_whilst_outputting_PRG

The ATC was in the process of sending the members of a PRG to the SMMU but before it had finished sending them, then a PRI Reset or signal reset occurred and the rest of the PRG will not be sent. The may leave a partial PRG in the PRI queue of the SMMU. Note the difference to having sent the entire PRG (including Last) and then receiving some sort of reset. Fields:

id unsigned int
: The ID of this request.

number_of_PRG_members unsigned int
: The number of members of the PRG, or 0 if run out of credits or PRGIndexes.

prgindex unsigned int
: The PRGIndex expected, or ~0u if run out of credits or PRGIndexes.

signal_reset bool
: True if it was a signal reset, false if it was a PRI Reset via the PRI Capability Header.

substreamid unsigned int
: The expected PASID on the _response_.

which_member_was_being_sent unsigned int
: The index of which member of the PRG was being sent whilst we received reset. All subsequent members of the PRG will not be sent.

ArchMsg.Warning.atc_pri_response_Response_Failure

A PRI Response has been received with a Response Failure. The PRI interface is required to disable itself, set RF in the header and will wake all waiting PRGs, forcing them to fault. Fields:

pasid unsigned int
: The PASID prefix on the response, or ~0u if none.

prgindex unsigned int
: The PRGIndex in the response.

response enum
: The response code

ArchMsg.Warning.atc_pri_response_ignored_as_RF_set

Any PRI Responses received whilst the RF (Response Failure) bit of the PRI header is required to be ignored by the PCIe specification. This is a response being ignored due to that reason. Fields:

pasid unsigned int
: The PASID prefix on the response, or ~0u if none.

prgindex unsigned int
: The PRGIndex in the response.

response enum
: The response code

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

ArchMsg.Warning.pri_enabled_with_zero_allocation

The PRI interface was enabled but the assigned credit allocation is zero. There is no useful way that the PRI interface can operate in this state and the model will fault any transaction that needs to use PRI. Fields:

capacity unsigned int
: The PRI Credit Capacity for this device.

ArchMsg.Warning.warning

These messages are about unusual (but not necessarily incorrect) activity occurring on the ATC. Fields:

output string
: The stream output

atc_commentary

A commentary from the cache about what is going on. Fields:

output string
: The stream output.

atc_entry

This monitors the formation or change of an atc entry. Fields:

atc_entry_id unsigned int
: An ID identifying this entry.

change_inconsistent_nug bool
: The change was inconsistent in NUG, and was ignored. This is almost certainly an error.

change_inconsistent_permissions bool
: The change was inconsistent in the permissions granted that were known before. The change was rejected for known changes. This may represent an error.

change_inconsistent_remap bool
: The input and/or output address ranges were changed. This may represent an error.

id unsigned int
: The ID of the transaction that caused this fill.

input_address_begin unsigned int
: The input address range covered by this entry.

input_address_end_incl unsigned int
: The input address range covered by this entry.

nugrwxrwx_change string
: The N, U, G and rwxrwx flags before and after any change.

output_address_begin unsigned int
: The output address range covered by this entry.

output_address_end_incl unsigned int
: The output address range covered by this entry.

reply_is_cacheable bool
: The reply was cacheable and so changed the known permissions.

substreamid unsigned int
: The SubstreamID of the entry.

atc_entry_allocated

An ATC entry has been allocated. Fields:

atc_entry_id unsigned int
: Index of ATC entry.

input_address_begin unsigned int
: The input address range covered by this entry.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

input_address_end_incl unsigned int
: The input address range covered by this entry.

nugrwxrwx string
: N, U, G bits and which permissions are unknown ('?'), granted (r/w/x) and denied ('-').

output_address_begin unsigned int
: The input address range covered by this entry.

output_address_end_incl unsigned int
: The input address range covered by this entry.

substreamid unsigned int
: The SubstreamID of the entry.

atc_entry_invalidated

An entry in the cache has been invalidated. Fields:

entry_id unsigned int
: The entry id that is being invalidated.

reason enum
: Why the entry is being invalidated.

atc_entry_none_formed

An ATC entry was attempted to be created but the ATS request forbade it being created. Fields:

fault enum
: Fault reason

id unsigned int
: The ID of the transaction that caused this fill.

input_address unsigned int
: The input address range covered by this entry.

substreamid unsigned int
: The SubstreamID that would have been used.

atc_inv_wait

An ATC Invalidate has to wait before completing because of outstanding transactions that are using the entry. Fields:

atc_inv_id unsigned int
: The ID of the ATC Invalidate transaction.

atc_inv_woke

An ATC Invalidate had to wait before completing because of outstanding transactions that are using the entry. This is it waking up. Fields:

atc_inv_id unsigned int
: The ID of the ATC Invalidate transaction.

because_of_reset bool
: The ATC Invalidate was killed because of reset.

atc_invalidate

An ATC Invalidate message was received. Fields:

Global_if_substreamid bool
: If has a SubstreamID then the Global flag.

aligned_address unsigned int
: The aligned address to invalidate.

atc_inv_id unsigned int
: The ID of this ATC Invalidate message.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

size_mask unsigned int
: The mask implied by the Size field.

substreamid unsigned int
: The SubstreamID (or ~0u if no SubstreamID)

atc_invalidate_kills_in_flight_ats

If an ATC Invalidate comes in whilst an ATS request is in flight that is affected by that ATC invalidate then it is killed and will be remade. Fields:

Global_if_substreamid bool
: If has a SubstreamID then the Global flag.

aligned_address unsigned int
: The aligned address to invalidate.

atc_id unsigned int
: The ID of the ATC that caused the ATS request.

atc_inv_id unsigned int
: The ID of this ATC Invalidate message.

size_mask unsigned int
: The mask implied by the Size field.

substreamid unsigned int
: The SubstreamID (or ~0u if no SubstreamID)

atc_pri_allocation_reduced_whilst_client_forming_prg

The client device was creating a PRG whilst the allocation for the PRI was changed. This left the PRI unable to satisfy the client device's request and we have to ask the client again what to do. Fields:

credits_used unsigned int
: The number of credits currently in use by in-flight PRGs.

new_allocation unsigned int
: The new allocation of credits set in the PRI header.

atc_pri_ats_transaction_failed

The ATS transaction failed to get the required permissions from the SMMU. PRI has been enabled so it now tries to form a PRI request. Fields:

allocation unsigned int
: The number of credits allocated to this device.

capacity unsigned int
: The advertised capacity of the device.

credits_used unsigned int
: The number of credits that are in use by in-flight PRG requests.

pri_enabled bool
: The PRI interface is enabled.

priv bool
: Privileged mode requested.

rwx enum
: Requested Read/Write/Execute permissions. Execute permission are only available for those requests with a PASID (SubstreamID), and read permission will automatically be asked for as well.

streamid unsigned int
: The StreamID of the request.

substreamid unsigned int
: The SubstreamID (PASID) of the request or ~0u if none.

trans_id unsigned int
: The transaction id of the ATS request.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

untranslated_address unsigned int
: The untranslated address of the ATS request.

atc_pri_client_device_chose_not_to_form_a_prg

The client device was given an opportunity to form a PRG and it chose not to. Instead the transaction will be faulted. Fields:

trans_id unsigned int
: The id of the failing ATS request.

atc_pri_could_not_form_prg_due_to_lack_of_prgindexes_or_credits

The client device was told that it couldn't form a PRG because of a lack of free PRGIndexes or because there were no free credits. The client device has the opportunity to either fault the transaction or to wait for some to become free. Fields:

trans_id unsigned int
: The id of the failing ATS request.

wait bool
: The client device chose to wait.

atc_pri_is_stopped

Traces why the PRI interface reports itself as stopped or not. Fields:

Stopped bool
: Is the PRI Stopped?

client_device_forming_requests bool
: The number of requests to the client device that are forming a PRG.

credits_outstanding bool
: The number of outstanding credits in outstanding PRGs.

atc_pri_members_of_prg

A client has formed a PRG and this describes each member of that PRG. Fields:

PRGIndex unsigned int
: The PRGIndex of this request.

member_index unsigned int
: The index of this member within the PRG.

number_of_members unsigned int
: The number of members in this PRG.

page_address unsigned int
: Page address of the request.

priv bool
: Privileged permissions requested. Valid only for those requests with a PASID (SubstreamID).

rwx enum
: Requested Read/Write/Execute permissions. Execute permission are only available for those requests with a PASID (SubstreamID), and read permission will automatically be asked for as well.

streamid unsigned int
: The StreamID of the PRI request.

substreamid unsigned int
: The SubstreamID sent with the PRG request.

trans_id unsigned int
: The transaction id of the failing ATS transaction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

atc_pri_outstanding_prg_request_killed_by_PRI_Reset_or_Response_Failure

This indicates that a PRI request (or one that was waiting for credits or a PRGIndexes to become free) was killed because either:- * software used PRI Reset in the PCIe PRI header, or * a PRI response was received with Response Failure, the effect of which is to kill all requests waiting for a PRG response. Fields:

trans_id unsigned int
: The transaction id of the ATS request that formed the PRI request.

atc_pri_outstanding_prg_when_interface_disabled

The PRI interface was disabled but there are outstanding PRI requests that have to be completed before the interface is stopped. Fields:

credits unsigned int
: The number of credits that this group uses.

expected_PASID_on_response unsigned int
: The expected PASID that should be on the PRI response (or ~0u if none expected).

prgindex unsigned int
: The PRGIndex of the outstanding request.

trans_id unsigned int
: The transaction id of the ATS request that formed the PRI request.

atc_pri_prg_request

A PRI Page Request Group (PRG) has. Fields:

PASID_on_response unsigned int
: The expected PASID on the response, or ~0u if none.

allocation unsigned int
: The number of credits allowed to be used.

credits_used unsigned int
: The number of credits in use now.

number_of_members unsigned int
: The number of members in the group.

prgindex unsigned int
: The PRGIndex used.

trans_id unsigned int
: The transaction id of the failing ATS transaction.

atc_pri_response_unrecognised_causes_uprgi

A PRI Response has been received that doesn't match a waiting PRG. The UPRGI bit in the PRI header will be set and then this response will be ignored. Fields:

match_PRG_currently_committing bool
: This is true if the PRI Response does match a PRG that is currently being sent to the SMMU, but because it is not yet complete then it should not have received a response yet and so the device treats this as UPRGI.

pasid unsigned int
: The PASID prefix on the response, or ~0u if none.

prgindex unsigned int
: The PRGIndex in the response.

response enum
: The response code

atc_pri_response_valid

A PRI Response has been received that is valid and we are going to wake up some threads of activity. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

credits_to_return unsigned int
: Credits that were used by the PRG and are now returned.

pasid unsigned int
: The PASID prefix on the response, or ~0u if none.

prgindex unsigned int
: The PRGIndex in the response.

response enum
: The response code

atc_pri_throwing_away_ppr

A client device asked for too many PRI Page Requests (PPR) in the PRG. The model is discarding PPRs to form the PRG. Fields:

page_address unsigned int
: Page address of the request.

priv bool
: Privileged permissions requested. Valid only for those requests with a PASID (SubstreamID).

rwx enum
: Requested Read/Write/Execute permissions. Execute permission are only available for those requests with a PASID (SubstreamID), and read permission will automatically be asked for as well.

streamid unsigned int
: The StreamID of the PRI request.

substreamid unsigned int
: SubstreamID (PASID), or ~0u if none.

trans_id unsigned int
: The transaction id of the failing ATS transaction.

atc_pri_waiting

Track a transaction that is waiting for a PRI request. Fields:

id unsigned int
: The ID of this request.

number_of_PRG_members unsigned int
: The number of members of the PRG, or 0 if run out of credits or PRGIndexes.

prgindex unsigned int
: The PRGIndex expected, or ~0u if run out of credits or PRGIndexes.

sleeping bool
: True if this is the trace of the transaction sleeping, false if it is waking.

substreamid unsigned int
: The expected PASID on the _response_.

atc_pri_woken_response

A transaction that was waiting for a PRI Response has been woken. Fields:

action enum
: The action now to take now woken.

id unsigned int
: The ID of this request.

prgindex unsigned int
: The PRGIndex expected, or ~0u if run out of credits or PRGIndexes.

substreamid unsigned int
: The expected PASID on the _response_.

control_write

This indicates a write to one of the control field of the ATC. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

comments enum
: Comments on the change

value_after_write unsigned int
: The value that the field was after the write.

which enum
: Which control field was written to

written_value unsigned int
: The value that was written

error_atc_entries_overlap

An ATC entry was attempted to be inserted into the ATC and was determined that it overlaps an existing entry. This check is not perfect but will catch simple errors. Fields:

end_address_of_new_entry unsigned int
: End address of new entry.

end_address_of_old_entry unsigned int
: End address of old entry.

index_of_new_entry unsigned int
: Index of new entry.

index_of_old_entry unsigned int
: Index of old entry.

start_address_of_new_entry unsigned int
: Start address of new entry.

start_address_of_old_entry unsigned int
: Start address of old entry.

final_remap_request

The final ATS result of the remap request. If this fails then PRI might be applied. In addition, PRI might make this remap request retry multiple times and is tracked by the retry_number. Fields:

address unsigned int
: Address of the request.

id unsigned int
: ID of ATC transaction.

inner enum
: The inner memory attribute.

instruction bool
: Is an instruction access.

is_debug bool
: Is a debug.

is_read bool
: Is a read request.

outer enum
: The outer memory attribute.

output_address unsigned int
: Output address.

privileged bool
: Is a privileged access.

result enum
: The result of the translation.

retry_number unsigned int
: The number of times this transaction has retried.

substreamid unsigned int
: Substream ID of the request (or ~0u if no SubstreamID).

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

initial_remap_request

The remap request for a bunch of transactions round this address. PRI might make this remap request retry multiple times and is tracked by the retry_number field. Fields:

address unsigned int
: Address of the request.

id unsigned int
: Unique ID of this request.

instruction bool
: Is an instruction access.

is_debug bool
: Is a debug.

is_read bool
: Is a read request.

privileged bool
: Is a privileged access.

retry_number unsigned int
: The number of times this transaction has retried.

substreamid unsigned int
: Substream ID of the request (or ~0u if no SubstreamID).

same_as

This request is the same as the specified other ATC transaction. Fields:

atc_entry_id unsigned int
: ID of the ATC Entry that it uses (or ~0ull if not relevant).

id unsigned int
: ID of ATC transaction.

same_as_id unsigned int
: ID of the ATC transaction it is the same as.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

4.124 v7_VGIC - trace

This section describes the trace sources.

CHECKPOINT_MESSAGE

Report error messages from the checkpointing process. Fields:

message string
: Message contents.

CHECKPOINT_RESTORE_END

Checkpoint restore completed.

CHECKPOINT_RESTORE_START

Checkpoint restore about to start.

CHECKPOINT_SAVE_END

Checkpoint save completed.

CHECKPOINT_SAVE_START

Checkpoint save about to start.

debug_out

Debug trace from the VGIC model, this contains implementation detail that is unlikely to be generally useful. Fields:

message string
: The message from the VGIC

sequence_id unsigned int
: Each successive message is labeled with an increasing number

gic_log_errors_out

Errors from the VGIC model. Fields:

message string
: The message from the VGIC

sequence_id unsigned int
: Each successive message is labeled with an increasing number

gic_log_fatal_out

Fatal error from the VGIC model. Fields:

message string
: The message from the VGIC

gic_log_warnings_out

Warnings from the VGIC model. Fields:

message string
: The message from the VGIC

sequence_id unsigned int
: Each successive message is labeled with an increasing number

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

vgic_cfgsdisable

The VGIC CFGSDISABLE pin has changed its state, this (un)locks down some configuration registers. Fields:

state bool
: The state of the signal

vgic_distributor_register_access

Distributor interface accesses on the VGIC. Fields:

cpu_id unsigned int
: The CPU that is making the request

data unsigned int
: The data in the access aligned to the LSB.

is_read enum
: Is the access a read?

offset enum
: Register within interface, top bit indicates NS state

offset_in_word unsigned int
: The offset of the access in the word.

size_in_bytes unsigned int
: The size of the access in bytes.

vgic_fiq_out

The FIQ output signal to the core. Fields:

cpu_id unsigned int
: The CPU id.

state bool
: The state of the signal

vgic_hypervisor_register_access

Hypervisor interface register accesses. Fields:

cpu_id unsigned int
: The CPU that is making the request

data unsigned int
: The data in the access aligned to the LSB.

is_read enum
: Is the access a read?

offset enum
: Register within interface, top bit indicates NS state

offset_in_word unsigned int
: The offset of the access in the word.

size_in_bytes unsigned int
: The size of the access in bytes.

vgic_irq_out

The IRQ output signal to the core. Fields:

cpu_id unsigned int
: The CPU id.

state bool
: The state of the signal

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

vgic_legacy_fiq_in

The legacy FIQ input signal into the VGIC. Fields:

cpu_id unsigned int
: The CPU id. There is one legacy IRQ/FIQ per core.

state bool
: The state of the signal

vgic_legacy_irq_in

The legacy IRQ input signal into the VGIC. Fields:

cpu_id unsigned int
: The CPU id. There is one legacy IRQ/FIQ per core.

state bool
: The state of the signal

vgic_physical_register_access

Physical interface accesses on the VGIC. Fields:

cpu_id unsigned int
: The CPU that is making the request

data unsigned int
: The data in the access aligned to the LSB.

is_read enum
: Is the access a read?

offset enum
: Register within interface, top bit indicates NS state

offset_in_word unsigned int
: The offset of the access in the word.

size_in_bytes unsigned int
: The size of the access in bytes.

vgic_reset

The reset pin on the VGIC has been changed. Fields:

state bool
: The state of the SPI signal

vgic_spi

Shared Peripheral Interrupt signal changed. Fields:

id unsigned int
: The interrupt id this will generate. SPIs start at ID 32 and so you will only see ids of >= 32

state bool
: The state of the SPI signal

vgic_vfiq_out

The VFIQ output signal to the core. Fields:

cpu_id unsigned int
: The CPU id.

state bool
: The state of the signal

vgic_virq_out

The VIRQ output signal to the core. Fields:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

cpu_id unsigned int
: The CPU id.

state bool
: The state of the signal

vgic_virtual_machine_register_access

Virtual machine interface register accesses. Fields:

cpu_id unsigned int
: The CPU that is making the request

data unsigned int
: The data in the access aligned to the LSB.

is_read enum
: Is the access a read?

offset enum
: Register within interface, top bit indicates NS state

offset_in_word unsigned int
: The offset of the access in the word.

size_in_bytes unsigned int
: The size of the access in bytes.

vgic_virtual_maintenance_interrupt

The VGIC is signaling that it needs software help from the hypervisor to maintain theVGIC interfaces. Fields:

cpu_id unsigned int
: The CPU id.

state bool
: The state of the signal

vgic_wakeup_fiq

The wakeup FIQ signal. Fields:

cpu_id unsigned int
: The CPU id.

state bool
: The state of the signal

vgic_wakeup_irq

The wakeup IRQ signal. Fields:

cpu_id unsigned int
: The CPU id.

state bool
: The state of the signal

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Chapter 5
Plug-ins for Fast Models

This chapter describes the plug-ins that are available for Fast Models.

Prebuilt plug-ins can be found at `$PVLIB_HOME/plugins/<OS_compiler>/`. The source code for some of these plug-ins is provided as programming examples, under `$PVLIB_HOME/examples/MTI/`.

It contains the following sections:

- *5.1 Loading a plug-in* on page 5-2891.
- *5.2 ArchMsgTrace* on page 5-2893.
- *5.3 BranchPrediction* on page 5-2894.
- *5.4 CADIIPCRemoteConnection* on page 5-2898.
- *5.5 CDE* on page 5-2899.
- *5.6 Crypto* on page 5-2909.
- *5.7 Fastline* on page 5-2911.
- *5.8 GDBRemoteConnection* on page 5-2919.
- *5.9 GenericCounter* on page 5-2921.
- *5.10 GenericTrace* on page 5-2922.
- *5.11 ListTraceSources* on page 5-2924.
- *5.12 PipelineModel* on page 5-2925.
- *5.13 RunTimeParameterTest* on page 5-2930.
- *5.14 ScalableVectorExtension* on page 5-2931.
- *5.15 Sidechannel* on page 5-2934.
- *5.16 TarmacText* on page 5-2935.
- *5.17 TarmacTrace* on page 5-2936.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

5.1 Loading a plug-in

The method of loading a plug-in depends on the type of model being used.

This section contains the following subsections:

- *5.1.1 --plugin command-line option* on page 5-2891.
- *5.1.2 scx::scx_load_plugin() method* on page 5-2891.
- *5.1.3 FM_TRACE_PLUGINS environment variable* on page 5-2891.

5.1.1 --plugin command-line option

For an FVP, or a SystemC EVS that uses the `scx::scx_parse_and_configure()` method, or when using Model Shell or Model Debugger, specify the plug-in using the `--plugin <path_to_plugin>/<plugin_name>.so` switch.

If more than one plug-in is required, use multiple `--plugin` switches.

Specify plug-in parameters using the `-C` option when launching the platform model, or in a configuration file.

For example, to load the `TarmacTrace` plug-in on Windows:

1. Run the following command to list both the model parameters and the plug-in parameters:

```
FVP_Base_AEMv8A.exe --plugin="%PVLIB_HOME%\plugins\Win64_VC2017\Release\TarmacTrace.dll"
--list-params
```

2. Run the following command:

```
FVP_Base_AEMv8A.exe -C bp.secure_memory=0 \
--plugin="%PVLIB_HOME%\plugins\Win64_VC2017\Release\TarmacTrace.dll" \
-C TRACE.TarmacTrace.trace-file=trace.log \
-a cluster0.cpu0=startup_AEMv8-FVP_AArch64_AC6.axf
```

   This command does the following:

   - Loads the `TarmacTrace` plug-in into the model using the model option `--plugin`.
   - Specifies the `TarmacTrace` log file using the plug-in parameter `TRACE.TarmacTrace.trace-file`.
   - Specifies the application to run on the `cluster0.cpu0` instance on the model, using the model option `-a`.

Related information

scx::scx_parse_and_configure

5.1.2 scx::scx_load_plugin() method

Fast Models that are exported to SystemC can call the method `scx::scx_load_plugin()` to hard code the path to the plug-in, before calling `sc_start()`.

For example:

```
scx::scx_load_plugin("$PVLIB_HOME/plugins/<OS_Compiler>/<plug-in_name>.so");
```

Related information

scx::scx_load_plugin

5.1.3 FM_TRACE_PLUGINS environment variable

When it is not possible to specify a trace plug-in to the launching tool, use the environment variable `FM_TRACE_PLUGINS`. This variable must be set to the full path of the plug-in.

For example, on Linux:

```
export FM_TRACE_PLUGINS=<installation_path>/plugins/<OS_Compiler>/TarmacTrace.so
```


Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

or on Windows:

```
set FM_TRACE_PLUGINS=<installation_path>\plugins\<OS_Compiler>\<version>\TarmacTrace.dll
```

To set multiple trace plug-ins at the same time, separate them with a semicolon, for example:

```
set FM_TRACE_PLUGINS=%PVLIB_HOME%\plugins\Win64_VC2017\Release\TarmacTrace.dll;%PVLIB_HOME%
\plugins\Win64_VC2017\Release\GenericTrace.dll
```

You can also load the same plug-in multiple times.

Give the plug-in instance a name by adding the prefix `<instancename>=` to the plug-in path or paths.

Specify parameters for plug-ins that you load using `FM_TRACE_PLUGINS`, using the following syntax:

```
|<param0_without_prefix=value>||<param1_without_prefix=value>||
<paramN_without_prefix=value>|<absolute_path_to_plugin.dll>
```

where `<param*_without_prefix=value>` means you must specify the parameter without the prefix that would be used on the command line. For example, use `trace-file=file.txt` instead of `TRACE.TarmacTrace.trace-file=file.txt`.

For example, on Linux:

```
export FM_TRACE_PLUGINS='|trace-file=/home/myname/work/trace.txt||end-instruction-count=1000|
$PVLIB_HOME/plugins/Linux64_GCC-6.4/TarmacTrace.so'
```

or on Windows:

```
set FM_TRACE_PLUGINS=^|trace-file=c:\work\trace.txt^|^|end-instruction-count=1000^|
%PVLIB_HOME%\plugins\Win64_VC2017\Release\TarmacTrace.dll
```

Note

- Do not specify a pipe character `|` at the end of the environment variable value.
- Separate individual parameters with two pipe characters `||`, not one.
- Separate the plug-in from the parameters using a single pipe character `|`.
- Do not specify quotes around paths that contain spaces, or escape spaces. Spaces are resolved automatically.
- Do not specify environment variables within this environment variable.
- Specify the parameters before you specify the plug-in.
- On Windows, you can either set the environment variable using the **Advanced System Settings** window, or if you are using the command line, you might need to escape each pipe character, for example `^|`.
- The parameter values that you set using this environment variable are not displayed by the `--list-params` or `-l` command-line options. This is because the environment variable is processed at a later stage, when launching the simulation.

Related concepts

1.12 Model Trace Interface on page 1-46

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

5.2 ArchMsgTrace

The Architecture Message plug-in prints warnings and error messages to `stdout` or to a file when the core executes unsafe or nonrecommended code.

This section contains the following subsection:

- *5.2.1 ArchMsgTrace parameters* on page 5-2893.

5.2.1 ArchMsgTrace parameters

This section describes the parameters for the ArchMsgTrace plug-in.

Each parameter is prefixed with `TRACE.ArchMsgTrace`, for example:

```
TRACE.ArchMsgTrace.suppress_sources
```

Table 5-1 ArchMsgTrace parameters

Parameter	Type	Allowed values	Default value	Runtime	Description
`exit_on_first_output`	`bool`	`true`, `false`	`false`	`false`	Exit the simulation process after the first message has been written.
`suppress_categories`	`string`	-	`"Why"`	`false`	Space-separated list of categories that should not be printed.
`suppress_repeated`	`bool`	`true`, `false`	`true`	`false`	Suppress repeated messages from similar call sites.
`suppress_sources`	`string`	-	-	`false`	Space-separated list of components or events that should not be printed.
`trace-file`	`string`	-	-	`false`	ArchMsgTrace output file.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

5.3 BranchPrediction

Fast Models supports branch prediction modeling by using the `BranchPrediction` plug-in.

Branch prediction is an aspect of Timing Annotation. For more details, see *Timing Annotation* in the *Fast Models User Guide*.

This section contains the following subsections:

- *5.3.1 BranchPrediction parameters* on page 5-2894.
- *5.3.2 BranchPrediction output* on page 5-2895.
- *5.3.3 Other ways to report branch mispredictions* on page 5-2895.
- *5.3.4 Create a branch predictor* on page 5-2896.

5.3.1 BranchPrediction parameters

This section describes the parameters for the BranchPrediction plug-in.

Each parameter is prefixed with `BranchPrediction.BranchPrediction`, for example:

```
BranchPrediction.BranchPrediction.bpstat-branchcount
```

Table 5-2 BranchPrediction parameters

Name	Type	Default value	Allowed values	Runtime	Description
`bpstat-branchcount`	`int`	`-0x1`	`0x0 - 0x7fffffffffffffff`	`false`	The number of branch data to display. Set to -1 to display all branch instructions.
`bpstat-pathfilename`	`string`	-	-	`false`	The path and filename of the branch statistics log, relative to the current directory. If you omit it, no log file is generated.
`mispredict-latency`	`int`	`0x8`	`0x0 - 0x7fffffffffffffff`	`false`	The number of instructions that are flushed for every misprediction. It is equal to the pipeline length of the processor. For example, in a processor where the prefetch unit has three stages and the execution units have five stages, the pipeline length is eight. Therefore the `mispredict-latency` is eight clock cycles.
`predictor-type`	`string`	`"CortexA53Predictor"`	See the note following the table.	`false`	The type of branch predictor to use.

Note

`predictor-type` can be one of the following example branch predictors that Arm provides, or a user-defined one:

`FixedDirectionPredictor`
An example predictor that always takes a preset fixed direction.

`BiModalPredictor`
An example of a standard 2-bit strength predictor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

GSharePredictor
An example of a standard global history sharing predictor.

HybridPredictor
An example predictor that selects the majority result from the bimodal predictor, fixed-direction predictor, and a random predictor.

CortexA53Predictor
Cortex-A53 branch predictor. This is the default.

5.3.2 BranchPrediction output

The following example command line loads the BranchPrediction plug-in into a model:

```
./EVS_Base_Cortex-A73x1 -a __image.axf \
…
--plugin $PVLIB_HOME/plugins/Linux64_GCC-6.4/BranchPrediction.so \
-C BranchPrediction.BranchPrediction.predictor-type=BiModalPredictor \
-C BranchPrediction.BranchPrediction.mispredict-latency=8 \
-C BranchPrediction.BranchPrediction.bpstat-pathfilename=bpstat.txt \
-C BranchPrediction.BranchPrediction.bpstat-branchcount=5
```

This command produces the following log file:

```
Processor Core: ARM_Cortex-A73
Cluster instance: 0
Core instance: 0
Mispredict Latency: 8
Image executed: __image.axf
PredictorType: BiModalPredictor
Total branch calls: 7757
Total Mispredictions: 130
Average prediction accuracy: 0.983241
Conditional Branches: 139
Total unique branch instructions: 289
--Branch instructions--
          PC Addr      Calls    Mispredict Accuracy
    [0]    0x0          2        0          1
    [1]    0x80000000   1        0          1
    [2]    0x8000000c   10       0          1
    [3]    0x80000014   10       0          1
    [4]    0x8000001c   10       0          1
```

5.3.3 Other ways to report branch mispredictions

These are some alternative ways to report branch mispredictions:

- The GenericTrace plug-in can generate MTI trace events that report branch mispredictions.

  Specify the following parameter to the `GenericTrace` plug-in:

  ```
  --plugin $PVLIB_HOME/plugins/Linux64_GCC-6.4/GenericTrace.so /
  -C TRACE.GenericTrace.trace-sources=BRANCH_MISPREDICT
  ```

  Note

  The `BranchPrediction` plug-in must also be loaded to generate these events.

- The `BranchPrediction` plugin is integrated with the PMU event counters. On a Linux boot simulation, you can track the number of simulated branch mispredictions in an application by loading the `BranchPrediction` plugin and running the `perf` tool. For example, the following command displays the number of branch mispredictions that are made in an application called `testapp`:

  ```
  perf stat –e branch-misses testapp
  ```


Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

5.3.4 Create a branch predictor

The example predictors that Arm provides are intended to show different techniques for branch prediction to help you create new ones. The source code for them is located in `$PVLIB_HOME/plugins/source/BranchPrediction/`.

A user-defined branch predictor must:

- Derive from the abstract interface class `PredictorInterface`.
- Implement the following functions:
  - `Direction getPredictedDirection(VAddr64_t pc_) SG_OVERRIDE;`

    Returns the predicted branch direction, that is, taken or not taken.
  - `void updateStructures(VAddr64_t pc_, Direction actual_direction_) SG_OVERRIDE;`

    Updates any tables or structures that the user-defined predictor implements, based on the actual branch direction.
- Implement the following static functions:
  - `static const char* getPredictorName()`

    Returns the name of the predictor. This is displayed in the log file.
  - `static PredictorInterface* createInstance()`

    Creates an instance of the predictor.
- Register the predictor in the `registerPredictors()` function, which is defined in `PredictorFactory.h`. For example:

```
void registerPredictors(){
      // All predictor types must be registered here
      map_predictor_instances[SamplePredictor::getPredictorName()] =
          &SamplePredictor::createInstance;
      …
}
```

Example 5-1 Branch predictor

```
// SamplePredictor.h
#include <ct/BranchPredictionPluginInterfaces.h>
#include "PredictorInterface.h"

namespace BranchPrediction
{
    class SamplePredictor : public PredictorInterface
    {
    public:
        Direction getPredictedDirection(VAddr64_t pc_) SG_OVERRIDE;
        void updateStructures(VAddr64_t pc_, Direction actual_direction_) SG_OVERRIDE;

    public:
        SamplePredictor(Direction dir_ = Direction::TAKEN);
        ~ SamplePredictor();

        static const char* getPredictorName() {
            return "SamplePredictor";
        }

        static PredictorInterface* createInstance() {
            return new SamplePredictor();
        }

    private:
        Direction default_direction;
    };
};


// SamplePredictor.cpp
#include "SamplePredictor.h"

namespace BranchPrediction
{
```


Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

```
    SamplePredictor::SamplePredictor()
    {
        // Initialize the predictor
    }

    SamplePredictor::~SamplePredictor()
    {
        // Do nothing
    }

    Direction SamplePredictor::getPredictedDirection(VAddr64_t pc_)
    {
        // Return the predicted direction
        // This is called when the CPU makes a branch prediction
        return Direction::TAKEN;
    }

    void SamplePredictor::updateStructures(VAddr64_t pc_, Direction actual_direction_)
    {
        // Update any structure or tables that this predictor uses
        // This is called when the branch has committed and the direction
        // taken is known.
    }
};
```


Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

5.4 CADIIPCRemoteConnection

Fast Models can be configured so that a remote CADI connection can be made to them.

The remote connection can be from a client on another machine, or a client on the same machine but using a specific port.

Explicitly enable remote connection support by adding the `CADIIPCRemoteConnection` command-line parameters. A CADI server must also be running, specified with `-S`.

This section contains the following subsection:

- *5.4.1 CADIIPCRemoteConnection parameters* on page 5-2898.

5.4.1 CADIIPCRemoteConnection parameters

This section describes the parameters for the `CADIIPCRemoteConnection` plug-in.

Each parameter is prefixed with `REMOTE_CONNECTION.CADIIPCRemoteConnection`, for example:

```
REMOTE_CONNECTION.CADIIPCRemoteConnection.range
```

Note

The default value restricts connections to be from the localhost only. To enable remote connections, specify an IP address to listen to, or specify `0.0.0.0` to listen to all adapters.

Table 5-3 CADIIPCRemoteConnection parameters

Name	Type	Default value	Allowed values	Runtime	Description
`enable_remote_cadi`	`bool`	`false`	`true`, `false`	`false`	Allow connections from remote hosts.
`listen_address`	`string`	`"127.0.0.1"`	`""`	`false`	If `enable_remote_cadi` is set, this parameter is the network address the server listens on.
`port`	`int`	`0x7b8b`	`0x1 - 0xffff`	`false`	If `enable_remote_cadi` is set, this parameter is the TCP port the server listens on.
`range`	`int`	`0x0`	`0x0 - 0x64`	`false`	If the requested port is not available, search for the next available port in the range [port:port+range]. The default is zero, which means only try the requested port.

Related information

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

5.5 CDE

Custom Datapath Extension (CDE) allows you to improve performance and efficiency by adding application domain-specific features to embedded processors, while maintaining the advantages of the Arm software ecosystem.

CDE allows you to add a customizable module inside an Arm Cortex-M33 or Cortex-M55 processor. This module is driven by the pre-decoded CDE instructions and shares the same interface as the standard Arithmetic Logic Unit (ALU) of the processor.

Fast Models implements CDE using Model Trace Interface (MTI) plug-ins, with CADI parameters to allow the plug-ins to be configured at runtime.

The following model parameters are exposed for configuring CDE:

`has_cde`
: Controls whether CDE is enabled. If enabled, a plug-in must be provided.

`--plugin` *`path/to/plugin.so`*
: This option can be specified multiple times, once for each CDE plug-in implementation. Alternatively, plug-ins can be loaded by setting the `FM_PLUGINS` or `FM_TRACE_PLUGINS` environment variable.

`cpu.cde_impl_name=`*`plugin_name`*
: The CDE implementation name to use with this core. If multiple CDE plug-in implementations are provided, each core can be requested to use a specific plug-in by using `cpu<`*`n`*`>.cde_impl_name=`….

Two example plug-ins are available, CDETester and CDEConstant. They are provided as pre-built libraries and as source code, located in `$PVLIB_HOME/plugins/source/`, from which you can derive your own plug-ins.

This section contains the following subsections:

- *5.5.1 CDETester* on page 5-2899.
- *5.5.2 CDETester parameters* on page 5-2900.
- *5.5.3 CDEConstant* on page 5-2902.
- *5.5.4 CDEConstant parameters* on page 5-2902.
- *5.5.5 Implementing a CDE plug-in* on page 5-2902.
- *5.5.6 CDE API* on page 5-2904.

5.5.1 CDETester

CDETester is a basic example plug-in that allows you to specify at runtime which CDE instructions are supported by individual coprocessors and to specify the behavior, either NOP or UNDEFINED.

To specify the instructions that coprocessors support, provide a plug-in parameter of the form:

```
CDE.CDETester.cde_tester_trivial.cps_implemented_instr=0xn
```

where `0x`*`n`* represents a hexadecimal bitmask of coprocessors that implement this instruction and *`instr`* represents a CDE instruction name.

The full list of CDE instruction names is as follows, where `d` represents dual variants and `v` represents vector variants:

- `cx1`
- `cx2`
- `cx3`
- `cx1d`
- `cx2d`
- `cx3d`
- `vcx1`

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

- vcx2
- vcx3
- vcx1v
- vcx2v
- vcx3v

Accumulate variants are handled in the same function implementation as the non-accumulate variants.

The bitmask takes the form:

bits [7:0]
For non-accumulate variants.

bits [23:16]
For accumulate variants.

The plug-in also allows control over which coprocessors implement CDE through the `CDE.CDETester.cde_tester_trivial.cps_implemented=0x`*n* parameter, where `0x`*n* represents a hexadecimal bitmask of coprocessors that implement CDE.

An example invocation to enable `CX1` and `CX1A` (accumulate) for CP3 might be written as:

```
CDE.CDETester.cde_tester_trivial.cps_implemented=0x8 // enable CDE support for CP3
CDE.CDETester.cde_tester_trivial.cps_implemented_cx1=0x80008 // enable CX1 and CX1A on CP3 (behaves as nop
rather than undef)
```

On Linux, you can retrieve an up-to-date list of parameters from the model by using:

```
-l | grep -i cdetester
```

5.5.2 CDETester parameters

This section describes the parameters for the CDETester plug-in.

Each parameter is prefixed with `CDE.CDETester`, for example:

```
CDE.CDETester.cde_tester_trivial.cps_implemented_vcx1
```

Table 5-4 CDETester parameters

Name	Type	Default value	Allowed values	Runtime	Description
`cde_tester_trivial.cps_implemented`	`int`	`0xff00ff`	`0x0 - 0xffffffff`	`false`	Bitmask indicating coprocessors implemented.
`cde_tester_trivial.cps_implemented_cx1`	`int`	`0xff00ff`	`0x0 - 0xffffffff`	`false`	Coprocessor enables CX1 instructions ([23:16] for CX1A, [7:0] for CX1).
`cde_tester_trivial.cps_implemented_cx1d`	`int`	`0xff00ff`	`0x0 - 0xffffffff`	`false`	Coprocessor enables CX1D instructions ([23:16] for CX1DA, [7:0] for CX1D).
`cde_tester_trivial.cps_implemented_cx2`	`int`	`0xff00ff`	`0x0 - 0xffffffff`	`false`	Coprocessor enables CX2 instructions ([23:16] for CX2A, [7:0] for CX2).

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 5-4 CDETester parameters (continued)

Name	Type	Default value	Allowed values	Runtime	Description
cde_tester_trivial.cps_implemented_cx2d	int	0xff00ff	0x0 - 0xffffffff	false	Coprocessor enables CX2D instructions ([23:16] for CX2DA, [7:0] for CX2D).
cde_tester_trivial.cps_implemented_cx3	int	0xff00ff	0x0 - 0xffffffff	false	Coprocessor enables CX3 instructions ([23:16] for CX3A, [7:0] for CX3).
cde_tester_trivial.cps_implemented_cx3d	int	0xff00ff	0x0 - 0xffffffff	false	Coprocessor enables CX3D instructions ([23:16] for CX3DA, [7:0] for CX3D).
cde_tester_trivial.cps_implemented_vcx1	int	0xff00ff	0x0 - 0xffffffff	false	Coprocessor enables VCX1 instructions ([23:16] for VCX1A, [7:0] for VCX1).
cde_tester_trivial.cps_implemented_vcx1v	int	0xff00ff	0x0 - 0xffffffff	false	Coprocessor enables VCX1 (Vector) instructions ([23:16] for VCX1A (Vector), [7:0] for VCX1 (Vector)).
cde_tester_trivial.cps_implemented_vcx2	int	0xff00ff	0x0 - 0xffffffff	false	Coprocessor enables VCX2 instructions ([23:16] for VCX2A, [7:0] for VCX2).
cde_tester_trivial.cps_implemented_vcx2v	int	0xff00ff	0x0 - 0xffffffff	false	Coprocessor enables VCX2 (Vector) instructions ([23:16] for VCX2A (Vector), [7:0] for VCX2 (Vector)).
cde_tester_trivial.cps_implemented_vcx3	int	0xff00ff	0x0 - 0xffffffff	false	Coprocessor enables VCX3 instructions ([23:16] for VCX3A, [7:0] for VCX3).
cde_tester_trivial.cps_implemented_vcx3v	int	0xff00ff	0x0 - 0xffffffff	false	Coprocessor enables VCX3 (Vector) instructions ([23:16] for VCX3A (Vector), [7:0] for VCX3 (Vector)).
has_cde_tester_trivial	bool	true	true, false	false	Tells the model whether the CDETester plug-in is implemented.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

5.5.3 CDEConstant

This is an example plug-in that provides an implementation for every CDE instruction variant, for example [V]CX*n*, [V]CX*n*A, and [V]CX*n*D. Each instruction simply performs an XOR with some arguments and a constant.

To load this plug-in, use the following parameters when launching the model:

```
--plugin path/to/plugin/CDEConstant.so -C cpu.has_cde=1 -C cpu.cde_impl_name=CDE_CONSTANT
```

5.5.4 CDEConstant parameters

This section describes the parameters for the CDEConstant plug-in.

Each parameter is prefixed with `CDE.CDEConstant`, for example:

```
CDE.CDEConstant.has_cde_constant
```

Table 5-5 CDEConstant parameters

Name	Type	Default value	Allowed values	Runtime	Description
has_cde_constant	bool	true	true, false	false	Tells the model whether the CDEConstant plug-in is implemented.

5.5.5 Implementing a CDE plug-in

Fast Models supports prototyping of custom instructions through a modular plug-in system, which uses the Model Trace Interface (MTI) framework.

Multiple CDE plug-ins can be registered with the model and each core can be instructed which plug-in behavior to use. Run-time configuration of CDE plug-ins is performed using CADI parameters, although plug-in developers can use alternative approaches, for example configuration files.

This guide shows how to implement a basic MTI-based CDE plug-in, using the CDETester plug-in as an example. It is intended to be a quick start to plug-in development, and does not describe details about MTI. To learn more about MTI, see *Model Trace Interface Reference Manual*. The source code for CDETester can be found in `$PVLIB_HOME/plugins/source/CDETester/`.

A CDE plug-in performs three main tasks:

- MTI and CDE interface registration.
- Handling of parameters passed through the command line or CADI.
- Implementing CDE instructions.

Prerequisites

To build the CDE plug-in examples, you need the following:

- A compiler that matches the Fast Models build you are using, for example Linux64_GCC7.3.
- A recent version of CMake.
- An installation of the Fast Models package and libraries.

Plug-in registration

After a CDE plug-in has been loaded into the model through the `--plugin` argument, or the `FM_TRACE_PLUGINS` environment variable, it must register itself with the CDE interface registry using the MTI framework API.

The CDE interface registry is responsible for:

- Managing all loaded CDE plug-ins.
- Passing any parsed arguments to the relevant CDE plug-in.
- Assigning CDE plug-ins to cores, as requested by the model arguments.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

The CDE interface registry requires an interface name and version to be registered through MTI, as shown in `CDETester.cpp` and `CDETesterTrivialImpl.h`.

Plug-in parameters

After the plug-in has registered itself with the CDE interface registry, the model passes any parsed command-line parameters to the CDE plug-in they are associated with.

The parameters to the CDETester example plug-in allow you to specify which custom instructions result in a no operation (NOP) for each coprocessor. Any instructions that are not enabled result in an Undefined Instruction exception being raised.

The plug-in handles parameters that it receives from the model in `CDETester.cpp` by passing them through the `CDETesterFactory` interface to the handler implementation in `CDETrivialImpl::consumeParameter()`, defined in `CDETesterTrivialImpl.cpp`.

Related references

5.5.2 CDETester parameters on page 5-2900

Plug-in instruction handler interface

After plug-in registration and optional parameter handling, the plug-in should implement handlers for all available CDE instructions, even if it does not intend to implement custom functionality for all instruction variants.

To help you do this, Fast Models provides a utility header, `$PVLIB_HOME/include/ct/CDE/CDEInstHandlerInterface.h`. This header defines the `CDEInstHandlerInterface` class for handling CDE instruction calls, which plug-ins must inherit.

This interface declares pure virtual methods for each CDE instruction variant, for example dual and accumulator, as well as the structures containing decode information and call results. A typical signature follows this pattern:

```
CDEResult64 CDETrivialImpl::exec_cx2_d(const CX2DecodeInfo& decode_info, uint64_t rfd_val, uint32_t rn_val)
```

The CDETester example plug-in inherits this interface in `CDETesterBaseImpl.h` and implements the instructions in `CDETesterTrivialImpl.h` and `CDETesterTrivialImpl.cpp`.

Plug-in instruction implementations

Each CDE instruction is mapped to a single function definition, except for accumulate variants, which are handled by checking for the accumulate flag in the parameters passed to the instruction implementations.

The full list of instructions that a plug-in is expected to implement is given in *5.5.1 CDETester* on page 5-2899.

Each instruction implementation accepts as parameters:

- A structure containing decoded instruction opcode parameters, including register numbers and immediate. For example `CX1DecodeInfo`.
- The contents of registers specified in the instruction opcode.

Instruction implementations should return a result structure of varying size, for example `CDEResult32` or `CDEResult64`. This structure indicates whether the instruction is supported, and if so, the return value and the number of cycles taken for execution, which is used in trace and performance analysis.

If a plug-in needs to raise an Undefined Instruction exception for a particular instruction, it can simply return a default-initialized result structure.

For full details of the expected function declarations and parameter types, see the file `$PVLIB_HOME/include/ct/CDE/CDEInstHandlerInterface.h`.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Building the plug-in

During plug-in development, either modify the example `CMakeLists.txt` files provided with the CDE plug-ins to reflect any changes to the file structure, or alternatively, use your own build system.

To build the example plug-ins:

1. Run `CMake` on the root directory of the plug-in to generate the project file, for example a Makefile or Visual Studio solution.
2. Run `make`.

Related concepts

5.5.1 CDETester on page 5-2899

5.5.6 CDE API

Reference documentation for the CDE API.

The CDE API is defined in the following header files, which are located in `$PVLIB_HOME/include/ct/CDE/`:

- `CDEFactoryInterface.h`
- `CDEInstHandlerInterface.h`
- `CDERegistryInterface.h`

CDEFactoryInterface.h

Defines the interface for obtaining an instance of a `CDEInstHandlerInterface` for a specific core.

CDE::CDEFactoryInterface class

The factory interface for obtaining an instance of a `CDEInstHandlerInterface` for a specific core.

CDE::CDEFactoryInterface::instantiateCDEInstHandler()

Instantiate a `CDEInstHandlerInterface`.

Instantiate a `CDEInstHandlerInterface` for a given core. The caller owns the result.

```
virtual std::unique_ptr<CDEInstHandlerInterface> instantiateCDEInstHandler(std::string component_hierarchy) =
0;
```

`component_hierarchy`
: String representing the hierarchy of the current component.

CDE::CDEFactoryInterface::CDEImplName()

Name of the CDE implementation.

Return the name of the CDE implementation. A core can use this method to disambiguate multiple CDE implementations in a simulation.

```
virtual std::string CDEImplName() const = 0;
```

CDE::CDEFactoryInterface::CDEImplDescription()

Description of the CDE implementation.

Return the description of the CDE implementation that can be instantiated.

```
virtual std::string CDEImplDescription() const = 0;
```

CDE::CDEFactoryInterface::CDEImplProviderName()

Provider name of the CDE implementation.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Return the name of the component providing the CDE implementation, for example a plug-in. This name might be used in informative diagnostic messages.

```
virtual std::string CDEImplProviderName() const = 0;
```

CDEInstHandlerInterface.h

Defines the interface for executing CDE instructions.

CDE::CDEResult32 struct

32-bit result of a CDE instruction.

Members

instr_not_supported
: Whether the instruction is supported.

value
: Return value of the instruction.

cycles
: Number of cycles of instruction execution.

CDE::CDEResult64 struct

64-bit result of a CDE instruction.

Members

instr_not_supported
: Whether the instruction is supported.

value
: Return value of the instruction.

cycles
: Number of cycles of instruction execution.

CDE::CDEResult128 struct

128-bit result of a CDE instruction.

Members

instr_not_supported
: Whether the instruction is supported.

value_lo
: Low 64 bits of the return value of the instruction.

value_hi
: High 64 bits of the return value of the instruction.

cycles
: Number of cycles of instruction execution.

CDE::CX1DecodeInfo struct

Decoded information for the `CX1` instruction.

Members

accumulate
: Whether to accumulate with existing register contents.

coproc
: Number of coproc.

imm
: Immediate value.

rd_num
: General-purpose destination register number.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CDE::CX2DecodeInfo struct

Decoded information for the `CX2` instruction.

Members

`accumulate`
: Whether to accumulate with existing register contents.

`coproc`
: Number of coproc.

`imm`
: Immediate value.

`rd_num`
: General-purpose destination register number.

`rn_num`
: General-purpose source register number.

CDE::CX3DecodeInfo struct

Decoded information for the `CX3` instruction.

Members

`accumulate`
: Whether to accumulate with existing register contents.

`coproc`
: Number of coproc.

`imm`
: Immediate value.

`rd_num`
: General-purpose destination register number.

`rn_num`
: General-purpose source register number.

`rm_num`
: General-purpose source register number.

CDE::VCX1DecodeInfo struct

Decoded information for the `VCX1` instruction.

Members

`accumulate`
: Whether to accumulate with existing register contents.

`coproc`
: Number of coproc.

`imm`
: Immediate value.

`vd_num`
: Source and destination vector register number.

CDE::VCX2DecodeInfo struct

Decoded information for the `VCX2` instruction.

Members

`accumulate`
: Whether to accumulate with existing register contents.

`coproc`
: Number of coproc.

`imm`
: Immediate value.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

vd_num
Source and destination vector register number.

vm_num
Source vector register number.

CDE::VCX3DecodeInfo struct

Decoded information for the VCX3 instruction.

Members

accumulate
Whether to accumulate with existing register contents.

coproc
Number of coproc.

imm
Immediate value.

vd_num
Source and destination vector register number.

vn_num
Source vector register number.

vm_num
Source vector register number.

CDE::CDEInstHandlerInterface class

Interface for executing CDE instructions.

This class defines the following methods for executing CDE instructions:

exec_cx1()	CX1 instruction.
exec_cx1_d()	CX1D instruction.
exec_cx2()	CX1 instruction.
exec_cx2_d()	CX2D instruction.
exec_cx3()	CX3 instruction.
exec_cx3_d()	CX3D instruction.
exec_vcx_1_s()	VCX1 instruction with S register.
exec_vcx_1_d()	VCX1 instruction with D register.
exec_vcx_1_q()	VCX1 instruction with Q register.
exec_vcx_2_s()	VCX2 instruction with S register.
exec_vcx_2_d()	VCX2 instruction with D register.
exec_vcx_2_q()	VCX2 instruction with Q register.
exec_vcx_3_s()	VCX3 instruction with S register.
exec_vcx_3_d()	VCX3 instruction with D register.
exec_vcx_3_q()	VCX3 instruction with Q register.
exec_vcx_1_beatwise()	VCX1 instruction for one beat. Caller handles predicated writeback.
exec_vcx_2_beatwise()	VCX2 instruction for one beat. Caller handles predicated writeback.
exec_vcx_3_beatwise()	VCX3 instruction for one beat. Caller handles predicated writeback.

CDE::CDEInstHandlerInterface::getCDECoprocessorMask()

Return a bitmask indicating which coprocessor numbers this CDE implementation subsumes.

```
virtual uint8_t getCDECoprocessorMask() = 0;
```


Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

CDERegistryInterface.h

Defines the interface to allow components, for instance plug-ins, to contribute CDE implementations to the simulation.

CDE::CDERegistryInterface class

Interface to register the CDE factory.

```
class CDERegistryInterface : public eslapi::CAInterface
```

This class is the interface to register the CDE factory into the Fast Models simulation component registry.

CDE::CDERegistryInterface::registerCDEFactory()

Register the CDE factory with the simulation.

```
virtual bool registerCDEFactory(std::ostream& error_stream, CDEFactoryInterface* interface) = 0;
```

`error_stream`
: The error stream.

`interface`
: The CDE factory interface used to register.

CDE::CDERegistryInterface::unregisterCDEFactory()

Unregister the CDE factory from the simulation.

```
virtual void unregisterCDEFactory(CDEFactoryInterface* interface) = 0;
```

`interface`
: The CDE factory interface used to unregister.

CDE::CDERegistryInterface::sendToCores()

Instantiate a `CDEInstHandler` and send it to the core.

The core can then call the CDE instruction execution functions provided by the plug-in on that `CDEInstHandler`.

```
virtual bool sendToCores() = 0;
```


Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

5.6 Crypto

The `Crypto` plug-in enables Armv8 processor models to support the Armv8.0 Cryptographic Extensions and Armv8.3 architected Pointer Authentication algorithms.

When the plug-in is loaded:

- All Armv8 processors in the system implement all functionality from the Armv8.0 Cryptographic Extensions by default, although you can disable it by setting the `CRYPTODISABLE` parameter for the core.
- All Armv8.3 processors in the system implement the architected algorithms for Pointer Authentication and Generic Authentication by default. Plug-in parameters are provided to control which processors in the system have architected algorithms enabled.

For AEMs, some parameters are available that allow you to restrict the `Crypto` plug-in features. These parameters use the same encodings as the flags within the AArch32 `ID_ISAR5` and AArch64 `ID_AA64ISAR0_EL1` system registers. You can set these parameters for a specific AEM core using this syntax:

```
-C cpu.cpu<X>.<feature_name>=<value>
```

Where *`feature_name`* can be one of the following:

- `crypto_aes`, with these possible values:
  - **0** No AES instructions are implemented
  - **1** The `AESE`, `AESD`, `AESMC`, and `AESIMC` instructions are implemented
  - **2** As 1, but in addition, the `PMULL` and `PMULL2` instructions can operate on 64-bit data values. This is the default value.
- `crypto_sha1`, with these possible values:
  - **0** No SHA-1 instructions are implemented
  - **1** The `SHA1C`, `SHA1P`, `SHA1M`, `SHA1H`, `SHA1SU0`, and `SHA1SU1` instructions are implemented. This is the default value.
- `crypto_sha256`, with these possible values:
  - **0** No SHA-256 instructions are implemented
  - **1** The `SHA256H`, `SHA256H2`, `SHA256SU0`, and `SHA256SU1` instructions are implemented. This is the default value.
- `crypto_sha3`, with these possible values:
  - **0** No Armv8.4 SHA-3 instructions are implemented. This is the default value.
  - **1** SHA-3 instructions are implemented if Armv8.4 is enabled.
  - **2** SHA-3 instructions are implemented.
- `crypto_sha512`, with these possible values:
  - **0** No Armv8.4 SHA-512 instructions are implemented. This is the default value.
  - **1** SHA-512 instructions are implemented if Armv8.4 is enabled.
  - **2** SHA-512 instructions are implemented.
- `crypto_sm3`, with these possible values:
  - **0** No Armv8.4 SM-3 instructions are implemented. This is the default value.
  - **1** SM-3 instructions are implemented if Armv8.4 is enabled.
  - **2** SM-3 instructions are implemented.
- `crypto_sm4`, with these possible values:
  - **0** No Armv8.4 SM-4 instructions are implemented. This is the default value.
  - **1** SM-4 instructions are implemented if Armv8.4 is enabled.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

2 SM-4 instructions are implemented.

Note

These parameters are only available for AEMv8-A core models. For other Armv8-A models, the behavior is fixed to the default values.

For example to disable the AES instructions on core 0:

```
./isim_system --plugin Crypto.so -C cpu.cpu0.crypto_aes=0
```

This section contains the following subsection:

- *5.6.1 Crypto parameters* on page 5-2910.

5.6.1 Crypto parameters

This section describes the parameters for the `Crypto` plug-in.

The parameters are prefixed with `CRYPTO.Crypto`, for example:

```
CRYPTO.Crypto.verbose=1
```

Table 5-6 Crypto parameters

Name	Type	Default value	Runtime	Description
`generic_authentication_core_pattern`	`string`	`*`	`false`	Install the Armv8.3 architected Generic Authentication algorithm on Armv8.3 cores that match one of these patterns. [t]
`pointer_authentication_core_pattern`	`string`	`*`	`false`	Install the Armv8.3 architected Pointer Authentication algorithm on Armv8.3 cores that match one of these patterns. [t]
`verbose`	`int`	`0x0`	`false`	Verbosity level: **0** Terse **1** Verbose

[t] The value is a comma-separated list of glob-style patterns which are matched against the instance names of cores in the system. If there is a match, the architected algorithm is installed on the core, if it supports Armv8.3. For example:

`*`	Install the algorithm on all supported cores
`cluster0.*`	Install the algorithm on all cores in cluster0
`cluster0.cpu0`	Install the algorithm only on cluster0.cpu0
`cluster0.*,cluster1.cpu1`	Install the algorithm on all cores in cluster0 and also on cluster1.cpu1
Empty string	Do not install the algorithm on any core

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

5.7 Fastline

Fastline is an MTI plug-in that enables you to use Streamline to analyze code running on Fast Models. The plug-in filename is `FastlineTrace.so` or `FastlineTrace.dll`.

Note

Fastline is deprecated from Fast Models release 11.12.

Fastline generates traces in `.apc` format that you can import into Streamline for analysis. It enables analysis of Linux-based kernels, native applications running on a Linux-based OS, and bare-metal applications that are built with debug information, without the need for extra instrumentation.

Fastline needs to be set up differently depending on whether it is running on a bare-metal or Linux target. Before you load the plug-in into a model, you must run one of the following Python scripts:

- For Linux, run `extract-offsets.py` to extract the address offsets of Linux kernel data structures.
- For bare-metal, run `process-elf.py` to generate a csv file that contains function addresses and symbol names.

These scripts are located in `$PVLIB_HOME/plugins/source/FastlineTrace/scripts/`.

Fastline has the following features:

- Allows you to set the sampling frequency. Increasing this value improves the accuracy of the trace, at the cost of increased file sizes and reduced simulation speed.
- Traces Linux thread context switches and `open()` and `execve()` syscalls.
- Works on SMP and multi-cluster systems.
- Maps MTI trace sources to Streamline counters.
- Allows you to set conditions on MTI trace sources, based on their field values.

Fastline has the following limitations:

- It only supports Linux kernel versions that use the process control and memory management corresponding to Linux kernel version 4.5.

This section contains the following subsections:

- *5.7.1 Using Fastline* on page 5-2911.
- *5.7.2 Cluster configuration file format* on page 5-2913.
- *5.7.3 System configuration file format* on page 5-2914.
- *5.7.4 Counter configuration file format* on page 5-2915.
- *5.7.5 Fastline parameters* on page 5-2917.

5.7.1 Using Fastline

Setting up Fastline involves the following steps.

Prerequisites

- An Armv7 or Armv8 model.
- The `FastlineTrace.so` or `FastlineTrace.dll` plug-in.
- Python 3.*. This is needed to run the `extract-offsets.py` or `process-elf.py` scripts.
- The `fromelf` executable. This is available in an Arm Compiler or Arm Development Studio installation. `fromelf` is used by the `extract-offsets.py` and `process-elf.py` scripts. The `PATH` variable must include the path to `fromelf`. These scripts have been tested with `fromelf` version 6.3 and later.
- Extra prerequisites for Linux applications:
  - The `System.map` file for the Linux kernel.
  - A Linux kernel image and a vmlinux file that was built with the `CONFIG_DEBUG_INFO` kernel option enabled.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Procedure

1. This step applies to Linux only. Run `extract-offsets.py` using the syntax:

   `extract-offsets.py` *`path/vmlinux_file`* `{1|0}`

   Specify 1 for 64-bit Linux or 0 for 32-bit Linux.

   Example:
   For example:

   ```
   ./extract-offsets.py vmlinux.o 1
   ```

   ———— **Note** ————

   You might need to change the access permissions for the current directory because the script creates a temporary file there.

   ————————————

   Results: The parameter that the script outputs might look like the following example. You will need to specify this parameter in the cluster configuration file:

   ```
   "linux-
   offsets-64":"16,0,1160,18446744073709551615,1048,1224,1228,1648,16,760,16,8,0,32,24,8,0,8,
   160,152,32,16,8,0,8,16"
   ```

2. This step applies to bare-metal applications only. Run `process-elf.py` to generate a csv file that you will need to specify in the cluster configuration file. Use the syntax:

   `process-elf.py` *`elf_file`*

   Example:

   For example:

   ```
   ./process-elf.py __image.axf
   ```

3. Create a cluster configuration file for each cluster that you want to profile. This is a JSON file that provides the Fastline plug-in with information about the bare-metal application or Linux kernel that is running on the cluster, and parameters to configure Fastline. The fields that you need to include in the file depend on whether you are targeting Linux or bare-metal. See `$PVLIB_HOME/plugins/config/FastlineTrace/` for the following cluster configuration file examples:

   `v8a_cluster_config_sample.json`
   : is a Linux example.

   `v7m_cluster_config_sample.json`
   : is a bare-metal example.

4. Create a system configuration file. This is a JSON file that maps cluster configuration files to clusters. See `$PVLIB_HOME/plugins/config/FastlineTrace/system_config_sample.json` for an example.
5. Optionally, create a counter configuration file. This is a JSON file that defines the parameters for the capture and maps MTI trace sources to Streamline counters. If you omit this parameter, the capture contains an instruction count only. See `$PVLIB_HOME/plugins/config/FastlineTrace/sample_counters.json` for an example.
6. Run the model, specifying the following parameters:
   - The Fastline plug-in library. Use the `--plugin <`*`path_to_plugin`*`>/<`*`plugin_name`*`>.so` parameter.
   - The system configuration file. Use the `config` parameter, for example:

     ```
     -C TRACE.FastlineTrace.config=config/system_config_sample.json
     ```

   - Optionally, the counter configuration file. Use the `counters-config-file` parameter, for example:

     ```
     -C TRACE.FastlineTrace.counters-config-file=config/sample_counters.json
     ```


Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

- Optionally, the filename and output directory for the capture file that Fastline generates. Use the `output-directory` and `apc-capture` parameters, for example:

```
-C TRACE.FastlineTrace.output-directory=/output_directory \
-C TRACE.FastlineTrace.apc-capture=test_run
```

- The model parameters.

Example:

For example:

```
./SVP_Base_Cortex-A73x1 --plugin FastlineTrace.so \
…
-C TRACE.FastlineTrace.config=config/system_config_sample.json \
-C TRACE.FastlineTrace.counters-config-file=config/sample_counters.json \
-C TRACE.FastlineTrace.output-directory=/output_directory \
-C TRACE.FastlineTrace.apc-capture=test_run
```

Results: This command generates a capture in `/output_directory/`, called `test_run.apc`, which you can import into Streamline for analysis.

7. Add the directory containing the capture to Streamline using the Streamline **Analysis Data Locations** dialog.
8. Double-click on the imported capture in the **Streamline Data** view to open the **Analyze** dialog. Select the ELF images for Streamline to analyze.
   Results: Streamline analyzes the capture and displays the analysis report in the **Timeline** view.

Related concepts

5.1 Loading a plug-in on page 5-2891

Related references

5.7.4 Counter configuration file format on page 5-2915

5.7.3 System configuration file format on page 5-2914

5.7.2 Cluster configuration file format on page 5-2913

5.7.5 Fastline parameters on page 5-2917

5.7.2 Cluster configuration file format

The Fastline cluster configuration file is a JSON file that provides the Fastline plug-in with information about the bare-metal application or Linux kernel that is running on the cluster, along with other parameters that configure the capture.

Example cluster configuration files are located in `$PVLIB_HOME/plugins/config/FastlineTrace/`:

- `v8a_cluster_config_sample.json` is a Linux example.
- `v7m_cluster_config_sample.json` is a bare-metal example.

The file can contain the following fields:

Table 5-7 Cluster configuration file fields

Name	Type	Applies to	Description
`operating-system`	`string`	Both	Either `"none"` to configure the plug-in for bare-metal applications, or `"Linux"`. Mandatory.
`sampling-frequency`	`int`	Both	The number of instructions to execute between samples. This value controls the rate at which backtrace messages are generated. These messages are used to construct the **Call Paths** and **Functions** views in Streamline. It is recommended to set this to a value that is greater than 50000. Mandatory.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 5-7 Cluster configuration file fields (continued)

Name	Type	Applies to	Description
`callstack-depth`	`int`	Linux only	The maximum depth to which the call stack will be unwound when generating the call path. Optional. The default is 100.
`linux-offsets-`*nn*	`string`	Linux only	A list of offsets that is generated by the `extract-offsets.py` script, where *nn* is either 32 or 64. This field is mandatory in the configuration file for the cluster that Linux is running on.
`pageshift`	`int`	Linux only	The value of the `PAGE_SHIFT` macro that was used during the kernel build. The size of memory pages that is supported by the kernel is: `1 << PAGE_SHIFT` Mandatory.
`system-map-file`	`string`	Linux only	The `System.map` file for the Linux kernel that is running on the model. Mandatory.
`trace-linux-syscalls`	`bool`	Linux only	Whether or not to trace Linux syscalls. Optional. The default is `true`.
`trace-linux-threads`	`bool`	Linux only	Whether or not to trace Linux context switches and thread lifetimes. This must be `true` to display any process-related activity in the Streamline **Timeline** view. Optional. The default is `true`.
`app-name`	`string`	Bare-metal only	The name of the bare-metal application to be analyzed. Mandatory.
`app-proc-id`	`int`	Bare-metal only	The process ID of the bare-metal application. This value must be unique across the system. Streamline cannot distinguish between a process running on Linux and a bare-metal application running on a different cluster if they have the same ID. Mandatory.
`report-function-activity`	`bool`	Bare-metal only	Whether or not Fastline should add a Function activity chart to the **Timeline** view in Streamline. Optional. The default is `false`.
`symbol-info-files`	`string` or a list of strings.	Bare-metal only	Specifies the path and filename of the csv file that the `process-elf.py` script generates. This field is mandatory in the configuration file for the cluster that the bare-metal application is running on.

5.7.3 System configuration file format

The Fastline system configuration file is a JSON file that defines which cluster configuration file to use for each cluster. Fastline always requires a system configuration file, even if there is only one cluster.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

For an example system configuration file, see `$PVLIB_HOME/plugins/config/FastlineTrace/system_config_sample.json`.

The file contains a list of cluster specifications, each of which has the following fields:

Table 5-8 System configuration file fields

Name	Type	Description
`Subsystem_name`	`string`	The name of the subsystem component to map the cluster configuration file to. This string can end with a wildcard, *.
`Subsystem_config_file`	`string`	The path and filename of the cluster configuration file.

5.7.4 Counter configuration file format

The Fastline counter configuration file is a JSON file that maps MTI trace sources to Streamline counters and defines the counter parameters.

For an example counter configuration file, see `$PVLIB_HOME/plugins/config/FastlineTrace/sample_counters.json`.

The file contains a list of counter specifications, each of which can contain the following fields:

Table 5-9 Counter configuration file fields

Name	Type	Description
`StartTime`	`uint64_t`	The simulated time in nanoseconds at which trace is enabled for this counter. Optional. The default is zero.
`EndTime`	`uint64_t`	The simulated time in nanoseconds at which trace is disabled for this counter. Optional. The default is `0xffffffffffffffff`.
`Component`	`string`	The MTI trace source component. This field can end with a wildcard, to match multiple components. If this field contains an empty string, this is equivalent to specifying a wildcard, `"*"`. Mandatory.
`MTISrcName`	`string`	The name of the MTI trace source. Mandatory. **Note** To list all the MTI trace sources that are present in the model, run it with the `ListTraceSources` plug-in.
`Condition`	`string`	One or more conditions to apply to the fields in the trace source. The counter is only incremented if the condition is met. Each entity in the comparison string must be space-separated, and you can use logical operators to compare them. For example: `"Condition" : "( FIELD1 == FIELD2 ) || ( ( FIELD3 >= VALUE1 ) && ( FIELD4 != VALUE2 ) )",` Optional. The default is `"true"`.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 5-9 Counter configuration file fields (continued)

Name	Type	Description
`Title`	`string`	The title of the chart under which the counter is displayed in the **Timeline** view in Streamline. Mandatory.
`CounterName`	`string`	The counter name. This corresponds to the series name in the **Timeline** view in Streamline. Mandatory.
`Type`	`string`	A string that is used by Streamline to uniquely identify the counter. If this field is omitted, Fastline uses the `Title` + `CounterName` strings as the `Type`. You only need to set this field if the `Title` + `CounterName` string does not uniquely identify the counter. Optional.
`Display`	`string`	Determines how the event counts that were collected in a given time period are processed and displayed when the user zooms out. The possible values are: **`Average`** Averages the underlying data. For example, this value might be used for energy probe data. **`Maximum` or `Minimum`** Displays the maximum or minimum value of the underlying data, and maintains that value until a new value is received from the target. For example, this value might be used for a memory chart. **`Accumulate`** Sums all underlying values. For example, this value might be used for branch mispredictons. **`Hertz`** Accumulates the data and normalizes it to one second. For example, this value might be used for clock cycles. Mandatory.
`Units`	`string`	Defines the unit of measurement. For example IPS (instructions per second) for an instruction count. Streamline converts this to MIPS or KIPS, depending on the value of the counter and displays this value. Optional. The default is `"units"`.
`AverageSelection`	`bool`	Whether Streamline should display an average of the values for the selected time period. Optional. The default is `true`.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 5-9 Counter configuration file fields (continued)

Name	Type	Description
`PerCpu`	`bool`	Whether the counter should be reported per core. For example, in traces that were generated using the example counter configuration file, `sample_counters.json`, the instruction count can be expanded to show per-core values, while cache events cannot. The `PerCpu` value controls this behavior. Optional. The default is `false`.
`MessageInterval`	`uint64_t`	Controls the rate at which messages are generated for the counter. For example, if you specify a `MessageInterval` of 100, Fastline produces 1 message that says the event occurred 100 times, instead of 100 messages that each say it occurred once. Increasing this value helps to reduce the trace file size and improve simulation speed, at the cost of accuracy. If it is not specified, the message is emitted every time the event occurs. Optional. The default is 1.

5.7.5 Fastline parameters

This section describes the parameters for the Fastline plug-in.

Each parameter is prefixed with `TRACE.FastlineTrace`, for example:

```
TRACE.FastlineTrace.config
```

Note

The `trace-start-time` and `trace-end-time` parameters apply to every counter. You can also specify a `StartTime` and `EndTime` for individual counters in the counter configuration file. If so, the time period in which trace occurs for a counter is the overlap between the two periods specified.

Table 5-10 Fastline parameters

Name	Type	Default value	Allowed values	Description
`apc-capture`	`string`	`"fastmodels.apc"`	`""`	Name of the APC capture directory that Fastline generates. Optional.
`config`	`string`	`""`	`""`	The system configuration file. This JSON file defines which cluster configuration file to use for each cluster. Mandatory.
`counters-config-file`	`string`	`""`	`""`	The counter configuration file. This JSON file maps MTI trace sources to Streamline counters and sets the parameters for the capture. Optional. If omitted, no counters are generated in the capture.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 5-10 Fastline parameters (continued)

Name	Type	Default value	Allowed values	Description
`output-directory`	`string`	`"."`	`""`	Directory for the APC capture that Fastline generates. Optional. If omitted, the APC is created in the current directory.
`time-base`	`int`	`0x0`	`0x0 - 0x1`	Selects whether time stamps are based on simulated time or host time: **0** This value is the default. Time stamps are based on simulated time and therefore are deterministic. This is useful when the `trace-start-time` or `trace-end-time` parameters are used, or if a repeatable time logging result is required. **1** Time stamps are based on host time, and therefore are non-deterministic. This can be useful for performance benchmarking, but is generally not recommended.
`trace-start-time`	`int`	`0x0`	`0x0 - -0x1`	The simulated time in nanoseconds at which trace is enabled for Fastline. Optional. If omitted, the default is zero.
`trace-end-time`	`int`	`-0x1`	`0x0 - -0x1`	The simulated time in nanoseconds at which trace is disabled for Fastline. Optional. If omitted, the default is `0xffffffffffffffff`.

Related information

Arm Streamline Performance Analyzer User Guide

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

5.8 GDBRemoteConnection

The `GDBRemoteConnection` plug-in allows the model to be debugged using GDB. It is installed as part of the Third-Party IP add-on package.

Load the plug-in into a model using the following command-line options:

- `--allow-debug-plugin`, or the short version, `-D`.
- `--plugin`.

For example:

```
FVP_VE_Cortex-A15x1 --allow-debug-plugin --plugin GDBRemoteConnection.so
```

Then, a suitable GDB can be connected to the model using the GDB `target` command.

It supports the following operations:

- Connection to models that contain a single core cluster.
- Read and write of core registers.
- Read and write of memory.
- Run, stop, single step.
- Breakpoints.
- Connection to AArch32 and AArch64 models.

This section contains the following subsections:

- *5.8.1 GDBRemoteConnection parameters* on page 5-2919.
- *5.8.2 GDBRemoteConnection limitations* on page 5-2919.

5.8.1 GDBRemoteConnection parameters

This section describes the parameters for the `GDBRemoteConnection` plug-in.

Each parameter is prefixed with `REMOTE_CONNECTION.GDBRemoteConnection`, for example:

```
REMOTE_CONNECTION.GDBRemoteConnection.memory_access_size
```

Table 5-11 GDBRemoteConnection parameters

Name	Type	Default value	Allowed values	Runtime	Description
`core_selection`	`int`	0	0-128	`false`	The core to drive with GDB.
`listen_address`	`string`	"0.0.0.0"	-	`false`	Network address the server should listen on.
`memory_access_size`	`int`	1	1-8	`false`	The access size for memory reads/writes in bytes.
`port`	`int`	31627	1-65535	`false`	TCP port the server should listen on.
`shutdown_on_disconnect`	`bool`	`false`	`true`, `false`	`false`	Shut down the model when the client disconnects.

5.8.2 GDBRemoteConnection limitations

This section describes the limitations of the `GDBRemoteConnection` plug-in.

- Connections are only allowed to single-core simulations, not to multiprocessor or multicluster simulations.
- No tracepoint support.
- No parameter support.
- Memory view only shows the current memory space.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

- Disassembly only uses the current instruction set.
- Breakpoints can only be set on the current memory space.

You cannot use GDB when debugging software that uses semihosting. When a program tries to use semihosting with the GDB plug-in, the GDB debugger wrongly reports having hit a breakpoint:

```
Program received signal SIGTRAP, Trace/breakpoint trap.
```

When using the `GDBRemoteConnection` plug-in, start the simulation with the `-D` (or `-S`) flag. You cannot attach the `GDBRemoteConnection` plug-in to running simulations. It can cause segmentation faults on the simulation and the GDB client.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

5.9 GenericCounter

`GenericCounter` is an example MTI plug-in that prints to stdout the number of occurrences of a specific trace source, at the end of the simulation.

The source code for this plug-in is provided in `%PVLIB_HOME%\examples\MTI\GenericCounter\`.

Note

This plug-in only counts a single trace source. To count multiple trace sources, load the plug-in multiple times.

This section contains the following subsection:

- *5.9.1 GenericCounter parameters* on page 5-2921.

5.9.1 GenericCounter parameters

This section describes the parameters for the `GenericCounter` plug-in.

The parameter is prefixed with `TRACE.GenericCounter`:

```
TRACE.GenericCounter.trace-source
```

Table 5-12 GenericCounter parameters

Name	Type	Default value	Allowed values	Runtime	Description
`trace-source`	`string`	`"INST"`	`""`	`false`	The trace source that you want to count. For example, `BRA_DIR`.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

5.10 GenericTrace

`GenericTrace` is a flexible MTI plug-in that allows you to configure which events are traced, using a comma-separated list of trace sources. Output can be printed to a text file or to the console.

Specify the trace sources of interest as a parameter to the plug-in, for example:

```
./FVP_Base_AEMv8A \
--plugin $PVLIB_HOME/plugins/Linux64_GCC-6.4/GenericTrace.so \
-C TRACE.GenericTrace.trace-sources=EXCEPTION,EXCEPTION_RETURN
```

Note

To see a list of the available trace sources for each component in the model that provides trace, run the model with the ListTraceSources plug-in. See *5.11 ListTraceSources* on page 5-2924 for details.

If no trace sources are specified, `GenericTrace` by default traces all the instructions.

To specify trace sources that match a pattern, use * or ? wildcards, for example:

```
TRACE.GenericTrace.trace-sources=CACHE_*
```

To trace a specific component only, specify the path to it, for example:

```
TRACE.GenericTrace.trace-sources=FVP_Base_AEMv8A.cluster0.cpu1.*
```

To trace specific fields from a trace source, use a field mask. For example to trace only the `PC` field of the `INST` trace source, use:

```
TRACE.GenericTrace.trace-sources=INST=0x1
```

The source code for this plug-in is provided as a programming example in `%PVLIB_HOME%\examples\MTI\GenericTrace\source\`.

This section contains the following subsection:

- *5.10.1 GenericTrace parameters* on page 5-2922.

5.10.1 GenericTrace parameters

This section describes the parameters for the `GenericTrace` plug-in.

Each parameter is prefixed with `TRACE.GenericTrace`, for example:

```
TRACE.GenericTrace.stop_on_event
```

Table 5-13 GenericTrace parameters

Name	Type	Default value	Allowed values	Runtime	Description
`enabled`	`bool`	`true`	`true`, `false`	`true`	If set to true, tracing is enabled.
`flush`	`bool`	`false`	`true`, `false`	`true`	If set to true, the trace file is flushed after every event. This parameter has a performance impact but can be used to debug crashes.
`hide-fieldnames`	`bool`	`false`	`true`, `false`	`true`	Do not print field names when printing trace output.
`perf-period`	`int`	`0x0`	`0x0` - `0x7fffffffffffff`	`false`	Print performance information after every *N* instructions. Disabled if 0.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 5-13 GenericTrace parameters (continued)

Name	Type	Default value	Allowed values	Runtime	Description
print-timestamp	bool	false	true, false	true	Start each trace entry with the host time.
shorten-paths	bool	true	true, false	true	If set to true, the component paths of trace events are shortened by removing the common prefix. The minimal, non-ambiguous path suffix remains. For example, FVP_VE_Cortex_A7x1.cluster.cpu0.INST is shortened to INST. If all traced sources belong to the same components, no path is logged.
simulated-timestamp	bool	false	true, false	true	Start each trace entry with the simulated time.
start-icount	int	0x0	0x0 - 0x7fffffffffffff	false	Start tracing on a certain instruction count. Tracing starts up to 2048 instructions before this count.
stop-icount	int	0x7fffffffffffff	0x0 - 0x7fffffffffffff	false	Stop tracing on a certain instruction count. Tracing will stop up to 2048 instructions after this count.
stop_on_event	bool	false	true, false	false	Stops the simulation when any event is triggered.
trace-file	string	""	""	true	The trace file to write to. If STDERR, prints to stderr. If empty, prints to stdout.
trace-sources	string	"INST"	""	true	A comma-separated list of trace sources to be traced. A component path can be prepended, with the components separated by dots. Both the component path and the trace source name can contain the wildcards * and ?. A field mask as a number in hex or decimal format can be appended with =. For example, my.subsystem.core.cpu*.TRACE_SOURCE=0x08.
verbose	bool	false	true, false	true	Print debugging information.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

5.11 ListTraceSources

`ListTraceSources` is an MTI plug-in that displays a complete and self-documenting list of the trace sources that a model provides, without running the model.

The source code for this plug-in is provided as a programming example in `%PVLIB_HOME%\examples\MTI\ListTraceSources\source\`.

The plug-in prints output for each component in the model, either to stdout or to a file, for example:

```
…
Component (4) providing trace: FVP_VE_Cortex_A7x1.cluster.cpu0 (ARM_Cortex-A7, 11.4.60)
================================================================================
Component is of type "ARM_Cortex-A7"
Version is "11.4.60"
#Sources: 195

Source ASYNC_MEMORY_FAULT (Context ID Register write.)
    Field FAULT type:MTI_UNSIGNED_INT size:4 (Fault status in ESR format)
    Field VADDR type:MTI_SIGNED_INT size:8 (Virtual Address (or 0 if unavailable))
    Field PADDR type:MTI_UNSIGNED_INT size:8 (Physical Address (or 0 if unavailable))
…
```

This section contains the following subsection:

- *5.11.1 ListTraceSources parameters* on page 5-2924.

5.11.1 ListTraceSources parameters

This section describes the parameters for the `ListTraceSources` plug-in.

Each parameter is prefixed with `TRACE.ListTraceSources`, for example:

```
TRACE.ListTraceSources.print_components_only
```

Table 5-14 ListTraceSources parameters

Name	Type	Default value	Allowed values	Runtime	Description
`file`	`string`	`""`	-	`false`	File to write the list of trace sources to. The default is to write to the console.
`print_components_only`	`bool`	`false`	`true`, `false`	`false`	If true, the plug-in prints the trace component information only, not the trace sources or fields.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

5.12 PipelineModel

Use the `PipelineModel` plug-in to estimate the performance of workloads within a Fast Models environment.

The plug-in models first-order effects of microarchitecture components on the overall Cycles Per Instruction (CPI) value. Examples of such effects are data and structural hazards due to instruction dependencies.

The `PipelineModel` is implemented as a Fast Models plug-in. It processes instruction traces and injects cycle latencies into the simulation. The plug-in is non-intrusive to the functional accuracy of the simulation.

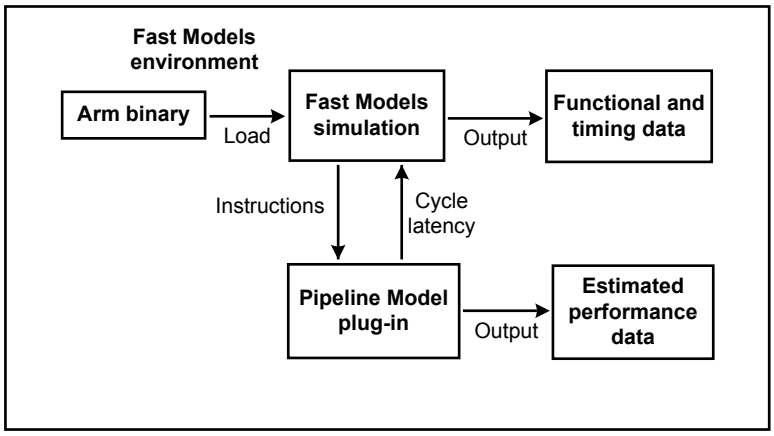

Figure 5-1 PipelineModel plug-in overview

Fast Models provides the following prebuilt example PipelineModel plugins:

CortexA53PipelineModel

An approximation of the performance characteristics of the in-order, dual issue properties of the Cortex-A53 processor. The model provides estimated performance characteristics of a given compute-bound workload in terms of Cycles Per Instruction.

Note

The model should not be used as a reference for hardware performance as it has limitations, such as the absence of a cache model.

InOrderPipelineModel

An implementation of a single-issue 4-stage pipelined processor that illustrates the basic components of a CPU. It demonstrates how components such as Fetch, Decode, and Execute can be implemented as a Fast Models plug-in.
The source code is provided in `%PVLIB_HOME%\plugins\source\PipelineModel\Cores\InOrder\`. It contains a README and a makefile for building the example.

Note

The model does not represent any Arm core and is intended only as a guide for developing more advanced PipelineModels using Fast Models.

The `PipelineModel` is an aspect of Timing Annotation. For more details, see *Timing Annotation* in the *Fast Models User Guide*.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

This section contains the following subsections:

- *5.12.1 PipelineModel parameters* on page 5-2926.
- *5.12.2 PipelineModel example* on page 5-2927.
- *5.12.3 Naming the plug-in instance* on page 5-2927.
- *5.12.4 Example command lines* on page 5-2927.
- *5.12.5 PipelineModel output* on page 5-2929.

5.12.1 PipelineModel parameters

This section describes the parameters for the `CortexA53PipelineModel` plug-in example. Other PipelineModel plug-ins might have different parameters.

Each parameter is prefixed with `PipelineModel` and the plug-in name, or plug-in instance name. For example:

```
PipelineModel.CortexA53PipelineModel.core-type
```

Table 5-15 PipelineModel parameters

Name	Type	Default value	Description
`core-type`	`string`	`""`	The name of the core. This value should match the platform that the plug-in is being attached to. For example: `ARM_Cortex-A53` This parameter is required.
`instance-name`	`string`	`""`	The cluster and core instance name, for example: `cluster0.cpu0` This parameter is optional. If it is specified, the plug-in only attaches to the specified instance. If it is not specified, the plug-in attaches to any instance.
`output`	`string`	`""`	The path and filename of the output statistics file, or `"stderr"`. This parameter is optional. If it is not specified, output is sent to `stdout`.
`start-pc`	`string`	`"0x00000000"`	The `PipelineModel` starts processing instructions from this PC value. If you want the `PipelineModel` to process only a subset of a program, use the `start-pc` and `end-pc` plug-in parameters. These values might represent the entry and exit of a function call, for example. After the `start-pc` address is hit, the `PipelineModel` processes instructions until one of the following occurs: • `end-pc` is hit. • A cycle limit, specified using the `--cyclelimit` model option, has elapsed. • The simulation is terminated. If `end-pc` is hit, the `PipelineModel` stops processing instructions and only resumes if `start-pc` is hit again. This parameter is optional. If it is not specified, or if `start-pc` and `end-pc` are the same, the plug-in processes all instructions.
`end-pc`	`string`	`"0x00000000"`	The `PipelineModel` stops processing instructions when this PC value is reached. This parameter is optional. If it is not specified, or if `start-pc` and `end-pc` are the same, the plug-in processes all instructions.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

5.12.2 PipelineModel example

This example shows how the `PipelineModel` plug-in generates data. This example uses a single issue, 6-stage pipeline. The stages are Fetch, Decode, Issue, EX1, EX2, and WR.

```
                         0   1   2   3   4   5   6   7   8   9   10
[0] ADD R1, R2, R3       IF  ID  IS  X0  X1  WR
[1] MUL R4, R1, R2           IF  ID  IS  IS  X0  X1  WR
[2] ADD R5, R4, R6               IF  ID  ID  IS  IS  X0  X1  WR
[3] ADD R7, R8, R9                   IF  IF  ID  ID  IS  X0  X1  WR
```

In the instruction sequence shown, pipeline stalls are underlined. The following stalls occur:

- Instruction [1] stalls one cycle in the Issue stage until R1 has been written.
- Instruction [2] stalls one cycle in the Decode stage due to a structural hazard because the Issuer is still in use by Instruction [1].
- Instruction [2] stalls one cycle in the Issue stage until R4 has been written.
- Instruction [3] stalls one cycle in the Fetch stage due to a structural hazard because the Decoder is still in use by Instruction [2].
- Instruction [3] stalls one cycle in the Decode stage due to a structural hazard because the Issuer is still in use by Instruction [2].

The `PipelineModel` plug-in uses the accumulated stalls for each instruction to calculate the runtime latency as follows:

- Instruction [0] latency = 0 cycles.
- Instruction [1] latency = 1 cycle.
- Instruction [2] latency = 2 cycles.
- Instruction [3] latency = 2 cycles.

In this example, there are five stall cycles in total. At the end of the simulation, the `PipelineModel` plug-in uses the accumulated stalls to produce a total cycle count. It uses the cycle count to determine the final Cycles Per Instruction (CPI) or Instructions Per Cycle (IPC) value.

5.12.3 Naming the plug-in instance

You can optionally assign a name to the plug-in instance. This is useful in a multiprocessor platform if you load the same plug-in multiple times. The assigned name is used to identify which plug-in instance the parameters apply to.

For example, the following commands load the `PipelineModel` plug-in twice and assign the names `CortexA53_0` and `CortexA53_1` to the first and second instances respectively:

```
--plugin CortexA53_0=CortexA53PipelineModel.so
--plugin CortexA53_1=CortexA53PipelineModel.so
```

You can then specify the plug-in instance name in the plug-in parameters. For example:

```
-C PipelineModel.CortexA53_0.core-type=ARM_Cortex-A53
-C PipelineModel.CortexA53_0.instance-name=Base.cluster0.cpu0

-C PipelineModel.CortexA53_1.core-type=ARM_Cortex-A53
-C PipelineModel.CortexA53_1.instance-name=Base.cluster0.cpu1
```

5.12.4 Example command lines

The following command lines show how to load the `PipelineModel` plug-in with a variety of platforms. Any parameters that are not relevant to the examples have been omitted.

Default usage, with no options

The following command line attaches the plug-in to any instance and sends output to `stdout`:

```
./EVS_Base_Cortex-A53x1.x
…
--plugin CortexA53PipelineModel.so
```


Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Single cluster, single processor platform

The following command line attaches the plug-in to `cluster0.cpu0`, which is a Cortex-A53 processor. It outputs the file `stat.txt`:

```
./EVS_Base_Cortex-A53x1.x
…
--plugin CortexA53PipelineModel.so
-C PipelineModel.CortexA53PipelineModel.core-type=ARM_Cortex-A53
-C PipelineModel.CortexA53PipelineModel.instance-name=Base.cluster0.cpu0
-C PipelineModel.CortexA53PipelineModel.output=stat.txt
```

Single cluster, multiprocessor platform, default usage

The following command line attaches the plug-in to all Cortex-A53 processors:

```
./EVS_Base_Cortex-A53x2.x
…
--plugin CortexA53PipelineModel.so
```

Single cluster, multiprocessor platform

The following command line loads two plug-in instances in a dual-processor platform. The first plug-in instance is named `CortexA53_0` and is attached to `cluster0.cpu0`. The second plug-in instance is named `CortexA53_1` and is attached to `cluster0.cpu1`:

```
./EVS_Base_Cortex-A53x2.x
…
--plugin CortexA53_0=CortexA53PipelineModel.so
--plugin CortexA53_1=CortexA53PipelineModel.so
-C PipelineModel.CortexA53_0.core-type=ARM_Cortex-A53
-C PipelineModel.CortexA53_0.instance-name=Base.cluster0.cpu0
-C PipelineModel.CortexA53_1.core-type=ARM_Cortex-A53
-C PipelineModel.CortexA53_1.instance-name=Base.cluster0.cpu1
```

Multicluster, single processor platform

The following command line names the plug-in instance `Core1` and attaches it to `cluster1.cpu0`:

```
./EVS_Base_Cortex-A73x1-A53x1.x
…
--plugin Core1=CortexA53PipelineModel.so
-C PipelineModel.Core1.core-type=ARM_Cortex-A53
-C PipelineModel.Core1.instance-name=Base.cluster1.cpu0
```

Plug-in core-type mismatch

The following command line specifies a Cortex-A55 processor. This mismatches the platform, which is Cortex-A53. The plug-in will fail to load:

```
./EVS_Base_Cortex-A53x1.x
…
--plugin CortexA53PipelineModel.so
-C PipelineModel.CortexA53PipelineModel.core-type=ARM_Cortex-A55
```

Plug-in instance-name mismatch

The following command line specifies an `instance-name` that does not exist. The plug-in will fail to load:

```
./EVS_Base_Cortex-A53x1.x
…
--plugin CortexA53PipelineModel.so
-C PipelineModel.CortexA53PipelineModel.instance-name=Base.cluster0.cpu2
```


Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Range processing

The following command line specifies an address range. The `PipelineModel` only processes instructions within this range:

```
-C PipelineModel.CortexA53PipelineModel.start-pc=0x80000001
-C PipelineModel.CortexA53PipelineModel.end-pc=8000FFFF
```

If the start and end of the range are identical, the `PipelineModel` processes all instructions. This is the same as not specifying a range at all:

```
-C PipelineModel.CortexA53PipelineModel.start-pc=0x80000001
-C PipelineModel.CortexA53PipelineModel.end-pc=0x80000001
```

Cycle limit

It might be useful to run the `PipelineModel` with a cycle limit. You can do this using the model option `--cyclelimit`. When the cycle limit is reached, the simulation terminates and the output file is generated.

The following command line causes the `PipelineModel` to process all instructions until the specified cycle limit is reached:

```
./EVS_Base_Cortex-A53x1.x
…
--cyclelimit=30000000
--plugin CortexA53PipelineModel.so
```

Note

The `--cyclelimit` option starts counting from the instruction at the beginning of the simulation, not from the instruction at `start-pc`. If the cycle limit has been reached and the `start-pc` has not yet been hit, the `PipelineModel` will not process any instructions.

5.12.5 PipelineModel output

At the end of the simulation, the `PipelineModel` generates a *Cycles Per Instruction* (CPI) value along with other performance data. You can use this for further analysis.

For example:

```
Elapsed time: 8 seconds
Instructions per second: 484186
Simulated CPU speed: 1.484186 MHz
CPU cycles: 5159984
RAW stalls: 3551018
Instructions issued: 3873492
Instructions retired: 3873491
Loads executed: 921403
Stores executed: 923306
IPC: 0.750679
CPI: 1.33213
```

The CPI value is the primary metric that measures the performance of workloads, where:

CPI = cycles elapsed / instructions retired

The lower the CPI, the better the performance. As a general indication, a CPI of 0.5 on a dual-issue, in-order or out-of-order processor means that an instruction takes 0.5 cycles to complete. In this case, the pipeline units are maximized and no latencies are generated.

Conversely, performance can be measured in *Instructions Per Cycle* (IPC), where:

IPC = instructions retired / cycles elapsed

The higher the IPC, the better the performance. As a general indication, an IPC of two on a dual issue, in-order or out-of-order processor means that on average, two instructions commit in each cycle.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

5.13 RunTimeParameterTest

`RunTimeParameterTest` is an example MTI plug-in that demonstrates how to add new string, integer, and boolean parameters at runtime.

This plug-in is provided only as source code, in `%PVLIB_HOME%\examples\MTI\RunTimeParameterTest\source\`.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

5.14 ScalableVectorExtension

The `ScalableVectorExtension` plug-in enables AEMv8-A and AEMvA models to support the Scalable Vector Extension (SVE) and SVE2.

Note

SVE and SVE2 are optional extensions to the Armv8 AArch64 architecture. Information about SVE is available in the *SVE supplement* to the Armv8-A Architecture Reference Manual. For a comparison of SVE and SVE2, see *SVE vs SVE2*.

This section contains the following subsection:

- *5.14.1 ScalableVectorExtension parameters* on page 5-2931.

5.14.1 ScalableVectorExtension parameters

This section describes the parameters for the `ScalableVectorExtension` plug-in.

Each parameter is prefixed with `SVE.ScalableVectorExtension`, for example:

```
SVE.ScalableVectorExtension.veclen
```

Note

To enable or disable support for particular features in the plug-in, use the `has_*` parameters.

Table 5-16 ScalableVectorExtension parameters

Name	Type	Default value	Runtime	Description
`clear_constrained_lanes`	`int`	`0x0`	`false`	When a constrained vector length increases, previously inaccessible bits are set to zero according to the value of this parameter. Possible values are: `0x0` Never `0x1` Always `0x2` If the register was written to while the vector length was constrained.
`combine_movprfx_and_destructive`	`bool`	`false`	`false`	Attempt to combine the execution of `MOVPRFX` and the destructively encoded instruction that follows it.
`disable_speculative_accesses`	`bool`	`false`	`false`	All speculative memory accesses behave as though faulting without accessing memory.
`enable_at_reset`	`bool`	`false`	`false`	Start with system registers set up for Scalable Vector Extension use.
`ffr_16b_pattern_UNKNOWN`	`int`	`0x0`	`false`	A specific 16-bit UNKNOWN value that is used by parameter `force_UNKNOWN_to_ffr`.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 5-16 ScalableVectorExtension parameters (continued)

Name	Type	Default value	Runtime	Description
force_UNKNOWN_to_ffr	int	0x0	false	Governs behavior if WRFFR writes a non-monotonic value to FFR. Possible values are: 0x0 Write non-canonical value to FFR. 0x1 Overwrite FFR with a specific 16-bit UNKNOWN value. See ffr_16b_pattern_UNKNOWN. 0x2 Clear all bits above first zero.
fp_exception_report_lowest	bool	false	false	If true, for multiple trapped FP exceptions, report the lowest lane in VECITR. Otherwise, report the highest.
fp_exception_set_tfv	bool	true	false	Set ESR_ELx.TFV during FP exception. Trapped exception flags are valid.
fp_exception_set_vecitr	bool	false	false	If true, set ESR_ELx.VECITR during FP exception. Otherwise, set RES0.
has_sve_bf16	bool	true	false	Whether SVE BFloat16 instructions are implemented.
has_sve_mm_f32	bool	true	false	Whether the SVE FP32 Matrix Multiply instructions are implemented.
has_sve_mm_f64	bool	true	false	Whether the SVE FP64 Matrix Multiply instructions are implemented.
has_sve_mm_i8	bool	true	false	Whether the SVE Int8 Matrix Multiply instructions are implemented.
has_sve2	bool	true	false	Whether SVE2 is implemented.
has_sve2_aes	int	0x2	false	If SVE2 is implemented, whether SVE2 AES instructions are implemented. Possible values are: 0x0 Not implemented. 0x1 SVE2 AESE, AESD, AESMC, and AESIMC are implemented. 0x2 Same as 1, but in addition, SVE2 PMULLB and PMULLT with 64-bit source are implemented.
has_sve2_bit_perm	bool	true	false	If SVE2 is implemented, whether BitPerm instructions are implemented.
has_sve2_sha3	bool	true	false	If SVE2 is implemented, whether SHA3 instructions are implemented.
has_sve2_sm4	bool	true	false	If SVE2 is implemented, whether SM4 instructions are implemented.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 5-16 ScalableVectorExtension parameters (continued)

Name	Type	Default value	Runtime	Description
movprfx_unpredictable_behavior	int	0x0	false	Defines the behavior of MOVPRFX and the instruction it immediately precedes when the behavior is CONSTRAINED UNPREDICTABLE. Possible values are: **0** UNDEF execution from MOVPRFX. **1** MOVPRFX and second half of instruction executes as NOP. **2** NOP MOVPRFX only.
predicated_sp_align_check_behaviour	int	0x0	false	Governs behavior of SP alignment checking for predicated memory accesses. Possible values are: 0x0 Always perform. 0x1 Skip if governing predicate is 0. 0x2 Skip for contiguous accesses if governing predicate is 0. 0x3 Skip for gather-scatter accesses if governing predicate is 0.
support_npot_vl	bool	true	false	Whether vector lengths that are not a power of two are supported.
undef_invalid_combined_movprfx	bool	true	false	If a combined MOVPRFX is invalid, raise an UNDEF exception. Otherwise NOP the second half of the register. This parameter is deprecated.
unknown_value	int	0xdeaddeaddeaddead	false	Simulated value for a state that has an UNKNOWN value after reset.
veclen	int	0x8	false	Size of the vector in units of 64-bit blocks. Allowed range is 0x2-0x20.
z_reg_on_load_fault_behaviour	int	0x0	false	Governs the behavior of destination Z-registers in case of a load fault. Possible values are: 0x0 Register becomes UNKNOWN. 0x1 Register is preserved.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

5.15 Sidechannel

The Generic Graphics Accelerator uses this plug-in for communication between the target and the host.

For more information, see *Graphics Acceleration in Fast Models* in the Fast Models User Guide.

This section contains the following subsection:

- *5.15.1 Sidechannel parameters* on page 5-2934.

5.15.1 Sidechannel parameters

This section describes the parameters for the Sidechannel plug-in.

Each parameter is prefixed with `DEBUG.Sidechannel`, for example:

```
DEBUG.Sidechannel.sh-diagnostics
```

Table 5-17 Sidechannel parameters

Name	Type	Default value	Allowed values	Runtime	Description
`diagnostics`	`int`	`0x0`	`0x0 - 0x3`	`false`	Diagnostic level: **0** None. **1** Calls. **2** With data dump.
`interceptor`	`string`	`-`	`-`	`false`	Interceptor `.so` to load.
`sh-diagnostics`	`int`	`0x0`	`0x0 - 0x3`	`false`	Diagnostic level for semihosting mechanism.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

5.16 TarmacText

TarmacText is an MTI plug-in that extracts the architectural execution trace, also known as Tarmac, of a processor. TarmacText extracts the trace in a textual form and saves it in a file. The trace that is produced conforms to the RTL execution trace.

The plug-in allows you to trace multiple processors simultaneously, saving the generated traces in different files.

Enable trace generation by setting `component` to the required component names, space-separated if multiple components are given. The default value of `component` is empty, which means the plug-in finds and traces all active processors.

Filenames are composed of a common prefix, configurable with the `log` parameter, followed by the name of the component, and terminated with the extension `.log`. The default value of `log` is `"tarmac"`.

Note

The platform name is trimmed from the component name.

This section contains the following subsection:

- *5.16.1 TarmacText parameters* on page 5-2935.

5.16.1 TarmacText parameters

This section describes the parameters for the TarmacText plug-in.

Each parameter is prefixed with `TRACE.TarmacText`, for example:

```
TRACE.TarmacText.start
```

Table 5-18 TarmacText parameters

Name	Type	Default value	Runtime	Description
`component`	`string`	`""`	`false`	Space-separated paths of the CPU instances to trace.
`log`	`string`	`"tarmac"`	`false`	Log tarmac text in a file named *<Log>*`.`*<component_name>*`.log`.
`start`	`int`	`0x0`	`false`	Start tracing after the specified number of steps. **Note** Tracing starts at the end of the quantum in which the count has been reached. Reducing the quantum size reduces the latency but might impact performance.
`use_instr_cnt_as_timestamp`	`int`	`0x0`	`false`	Use the instruction count as the timestamp instead of the simulation time.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

5.17 TarmacTrace

`TarmacTrace` is an MTI trace plug-in that prints information to `stdout` or to a file.

This information might include instructions, program flow, or memory accesses for ARMv7 or ARMv8 processors in the simulation. Plug-in parameters control the amount and type of information that is traced.

This section contains the following subsections:

- *5.17.1 TarmacTrace parameters* on page 5-2936.
- *5.17.2 TarmacTrace file format* on page 5-2939.
- *5.17.3 Instruction trace* on page 5-2939.
- *5.17.4 Program flow trace* on page 5-2941.
- *5.17.5 Register trace* on page 5-2941.
- *5.17.6 Cache maintenance trace* on page 5-2942.
- *5.17.7 Cache content trace* on page 5-2943.
- *5.17.8 Translation table walk trace* on page 5-2944.
- *5.17.9 TLB trace* on page 5-2945.
- *5.17.10 Event trace* on page 5-2946.
- *5.17.11 Processor memory access trace* on page 5-2948.
- *5.17.12 Processor memory update trace* on page 5-2948.
- *5.17.13 Memory bus trace* on page 5-2949.
- *5.17.14 Tarmac Trace output example* on page 5-2951.

5.17.1 TarmacTrace parameters

This section describes the parameters for the `TarmacTrace` plug-in.

Each parameter is prefixed with `TRACE.TarmacTrace`, for instance:

```
TRACE.TarmacTrace.end-instruction-count
```

Table 5-19 TarmacTrace parameters

Name	Type	Default value	Allowed values	Description
`end-instruction-count`	`int`	`0`	`0 - 0x7FFFFFFFFFFFFFFF`	The instruction count at which to stop tracing. Default is to never stop tracing.
`instruction-count-is-per-target`	`bool`	`true`	`true`, `false`	If true (default) then the `start-instruction-count` and `end-instruction-count` parameters apply to individual targets separately. If false, all components start and stop tracing when the first component reaches the instruction count.
`loadstore-display-width`	`int`	`8`	`0 - 40`	Memory transactions can involve up to 64 bytes. For easier readability, these transactions can be broken up into multiple memory access records with a smaller number of bytes. 0 means do not break up any transaction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 5-19 TarmacTrace parameters (continued)

Name	Type	Default value	Allowed values	Description
quantum-size	int	0x2710	0x1 - 0x7fffffff	Set the default quantum size that is used in computing when the tracing should start and stop. The CORE_INFO.QUANTUM_SIZE trace source field of the component overrides this parameter, when present.
quiet	bool	false	true, false	Limit output to trace information.
start-instruction-count	int	0	0 - 0x7FFFFFFFFFFFFFFF	The instruction count at which the tracing starts. Default is to trace from the beginning.
trace_aarch64_vfp_full_width	bool	false	true, false	Trace a write to an S or D register in AArch64 as a write to the corresponding V register.
trace_atomic	bool	true	true, false	Trace memory updates by atomic operations.
trace_branches	bool	false	true, false	Trace changes of the program flow like direct or indirect branches and exception returns.
trace_bte	bool	true	true, false	Trace opcodes rejected by BTE (Branch Target Enforcement) instructions.
trace_bus_accesses	bool	false	true, false	Trace bus accesses of the core, including accesses by the caches of the core. This trace slows down the model considerably.
trace_cache	bool	true	true, false	Trace cache fills and evictions.
trace_core_registers	bool	true	true, false	Trace the core registers R0-R14, the CPSR, and the SPSR.
trace_cp15	bool	true	true, false	Trace writes to CP15 registers.
trace_dap	bool	true	true, false	Trace accesses on the debug access port.
trace_ete	bool	true	true, false	Trace packets generated by the ETE.
trace_events	bool	true	true, false	Trace events, for example exceptions and mode changes.
trace_exception_reasons	bool	true	true, false	Trace INFO_EXCEPTION_REASONS (M-class processors only).
trace-file	string	""	-	Trace output file (stdout by default, stderr if "STDERR").
trace-file-per-comp	bool	false	true, false	Write trace to multiple files.
trace_generic_events	bool	false	true, false	Trace generic events.
trace_gicv3	bool	true	true, false	Trace GICv3 memory-mapped accesses.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 5-19 TarmacTrace parameters (continued)

Name	Type	Default value	Allowed values	Description
trace_gicv3_comms	int	0	0 - 7	Trace GICv3 communications between cores and distributor. Bitfield: **1** Trace processor. **2** Trace RD0. **4** Trace internal.
trace_gicv3_its	bool	false	true, false	Trace GICv3 ITS command execution.
trace_gicv3_reads	bool	false	true, false	Trace GICv3 memory-mapped reads.
trace-inst-stem	string	""	-	Base instance path to select an instance or group of instances to trace. For example: `-C TRACE.TarmacTrace.trace-inst-stem=css.cluster0.cpu3`
trace_instructions	bool	true	true, false	Trace instructions.
trace_loads_store_memtype	bool	false	true, false	Show memory type information for core loads and stores.
trace_loads_stores	bool	true	true, false	Trace loads and stores that are triggered by instructions. These loads and stores might go into the memory subsystem, into a cache, or into a TCM. This trace slows down the model considerably.
trace_memory	bool	false	true, false	Trace memory accesses just outside the core.
trace_mmu	bool	true	true, false	Trace MMU tablewalks and associated information.
trace_spe	bool	true	true, false	Trace SPE data that is written to memory.
trace_tag_loads_stores	bool	true	true, false	Trace tag loads and stores that are triggered by MTE instructions. These might go into the memory subsystem, into a cache, or into a TCM. This trace slows down the model considerably.
trace_vfp	bool	true	true, false	Trace the VFP and Neon registers, including FPSCR and FPEXC.
unbuffered	bool	false	true, false	Trace events as they arrive, and flush each `fwrite`. Prints `IT` even when `IS` should be printed.
updated-registers	bool	false	true, false	Trace updated value of registers rather than the written value.
use_instr_cnt_as_timestamp	bool	false	true, false	Use the instruction count as the timestamp instead of the simulation time.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

5.17.2 TarmacTrace file format

This section describes the TarmacTrace file format.

Note

[*X*|*Y*]
Indicates a choice between *X* and *Y*.

{*X*}
Indicates that *X* is optional or configuration dependent.

The common address definition that is used in the trace command syntax:

```
<vaddr>{:<paddr><psecurity>}
```

`<vaddr>`
The virtual address in hexadecimal format. See the note below.

`<paddr>`
The physical address of the instruction, if available, in hexadecimal. See the note below.

`<psecurity>`
Append `_NS` if the security regime of the physical address is Non-secure. Append nothing if the regime is Secure.

Note

For 64-bit addresses, the value is written as either:

- 8 hex digits, if the value can be represented in 32 bits.
- 16 hex digits otherwise.

The virtual regime definition that is used in the trace command syntax:

```
0x<vbase>{_NS} <el>{ vmid=<vmid>}{, nG asid=<asid>}
```

`0x<vbase>`
Virtual address in hexadecimal format.

`_NS`
If present, this element specifies that the address is Non-secure. If not present, the address is Secure.

`<el>`
Translation regime that owns the mapping. One of:

- `EL1_n`, meaning the Non-secure EL1&0 translation regime.
- `EL2_n`
- `EL1_s`
- `EL3_s`

`<vmid>`
For Non-secure, non-hyp regimes, specify the VMID.

`nG`
If present, this element specifies that the virtual regime is non-global.

`asid`
For non-global regimes, specify the ASID.

5.17.3 Instruction trace

If enabled, this trace source generates one record for every instruction started.

The records (lines) of the instruction trace have this syntax:

```
<time> <scale> <cpu> [IT|IS] (<inst_id>) <addr> <opcode> [A|T|X|O]
<mode>_<security> : <disasm>
```


Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

<time>

Timestamp (decimal value).

<scale>

Unit for <time>. clk indicates that the timestamp is not related to real time, but an increasing count.

<cpu>

Processor, or other component, that gave the instruction.

[IT|IS]

IT
Instruction passed the condition code (taken).

IS
Instruction failed the condition code (skipped).

<inst_id>

Tick count of this processor, which is equivalent to the number of instructions that are executed, except for certain instructions like WFI/WFE (decimal value).

<addr>

Fetch source address for this instruction. Format according to the common address definition.

<opcode>

16-bit/32-bit hexadecimal opcode of the instruction.

[A|T|X|O]

Instruction set:

A
A32.

T
T32.

X
T32EE.

O
A64

<mode>

Processor execution mode.

AArch32 modes are svc, irq, fiq, usr, mon, sys, abt, und, hyp.

AArch64 modes are EL3h, EL3t, EL2h, EL2t, EL1h, EL1t, EL0t.

<security>

Processor security state (s or ns).

<disasm>

Disassembly of the instruction.

Related references

5.17.2 TarmacTrace file format on page 5-2939

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

5.17.4 Program flow trace

If enabled, every executed branch instruction triggers this trace source, which is a more efficient way to reconstruct the program flow than by tracing every instruction.

Output syntax:

`<time> <scale> {<cpu>} [FD|FI] (<inst_id>) <addr> <targ_addr> [A|T|X|O]`

`<time>`

Timestamp (decimal value).

`<scale>`

Unit for `<time>`. Gives consistency with device-specific Tarmac Trace formats.

`<cpu>`

Processor, or other component, that gave the instruction.

`[FD|FI]`

Program flow change by:

`FD`
A direct branch.

`FI`
An indirect branch.

`<inst_id>`

Tick count of this processor, which is equivalent to the number of instructions that are executed, except for certain instructions like `WFI`/`WFE` (decimal value).

`<addr>`

Fetch source address for this instruction. Format according to the common address definition.

`<targ_addr>`

Address (virtual) at which the execution continues. Format according to the common address definition.

`[A|T|X|O]`

Instruction set after the branch:

`A`
A32.

`T`
T32.

`X`
T32EE.

`O`
A64

Related references
5.17.2 TarmacTrace file format on page 5-2939

5.17.5 Register trace

If enabled, this source traces all writes to the processor registers.

This trace source includes writes to core registers `R0` to `R14`, `X0` to `X30`, `CPSR`, and `SPSR`, VFP registers such as `S0` to `S31`, `D0` to `D31`, `FPSCR`, and `FPEXC`, and writes to system registers including `CP14`, `CP15`, and

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

GIC. Banked registers are traced separately using the mode as a suffix to the register name, for example r13 (current register R13) and r13_mon (banked register R13).

Output syntax:

```
<time> <scale> {<cpu>} R <register> <value>
```

<time>

Timestamp (decimal value).

<scale>

Unit for <time>, which gives consistency with device-specific Tarmac Trace formats.

<cpu>

Processor, or other component, that gave the instruction.

<register>

Register name in lowercase letters. Banked core registers can have a mode appended to them with a single underscore. Banked CP14/CP15 registers have _s or _ns appended to indicate access of either the Secure or Non-secure banked register.

<value>

Hexadecimal value that is written to the register (64 bits maximum).

If the SVE plug-in is loaded in the model, there are additional registers in the program view. The output examples below show how these registers are traced when the value changes. These data values can be very large.

```
8463 clk cpu0 IT (8439) 000282c0:0000152282c0_NS 053fc01f O EL1h_n : SEL
z31.B,p0,z0.B,z31.B
8463 clk cpu0 R z31 00000000_00000000_00000000_00000000
```

R indicates a register write. z0 to z31 are the vector registers. The written data are hexadecimal digits, which are separated by an underscore every 32 bits. The length of the written data varies with the configuration, depending on the vector length.

```
9756 clk cpu0 IT (9732) 01000074:000011000074_NS 2518e3e0 O EL1t_n : PTRUE    p0.B,ALL
9756 clk cpu0 R p0 ffff
```

R indicates a register write. p0 to p15 are the predicate registers. The written data are hexadecimal digits. If they are long enough to require one, the digits are separated by an underscore every 32 bits. The length of the written data varies with the configuration, depending on the vector length. Predicate registers contain 1 bit per byte of vector register length.

5.17.6 Cache maintenance trace

If enabled, this source traces all cache maintenance operations that the processor initiates.

Output syntax:

```
<time> <scale> <cpu> CACHE MAINTENANCE <side> <operation> <scope> <data> {<pagesize>
<memtype>}
```

<time>

Timestamp (decimal value).

<scale>

Unit for <time>. clk indicates that the timestamp is not related to real time, but an increasing count.

<cpu>

Processor, or other component, that gave the instruction.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

<side>
Data or instruction cache.

<operation>
Clean, invalidate, or both.

<scope>
By MVA or set/way, to Point of Coherency or Point of Unification, Inner Sharable or not.

<data>
Data that is associated with the operation. If the operation is by MVA, format according to the common address definition, otherwise use raw hexadecimal.

<pagesize>
If the operation is by MVA, this element is the size of the memory region that is described by the TLB entry which contains the MVA.

<memtype>
If the operation is by MVA, this element is the type of memory in the TLB entry which contains the MVA.

Related references
5.17.2 TarmacTrace file format on page 5-2939

5.17.7 Cache content trace

Traces the movement of data into and out of the cache.

Output syntax:

```
<time> <scale> <cpu> CACHE <id> LINE <line> <operation> 0x<paddr><ns>
```

<time>

Timestamp (decimal value).

<scale>

Unit for `<time>`. `clk` indicates that the timestamp is not related to real time, but an increasing count.

<cpu>

Processor, or other component, that gave the instruction.

<id>
Level and side, or system identifier, of the cache.

<line>
Identifier of this line uniquely within this cache, expressed in hexadecimal.

<operation>
Notification for this cache line. One of the following options:

`ALLOC`
(Processor caches) Line contains new read data.

`INVAL`
(Processor caches) Line contains no data.

`DIRTY`
(Processor caches) Line contains new write data.

`CLEAN`
(Processor caches) Write data is written back, still valid for reads.

`FILL`
(System caches) Line is filled.

`EVICT`
(System caches) Line is evicted due to space pressure.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

`CLEAN`
(System caches) Line is cleaned due to maintenance operation.
`INVAL`
(System caches) Line is invalidated due to maintenance operation.

`<paddr>`
Cache line physical address in hexadecimal.

`<ns>`
Cacheline security. Blank for Secure regime, or `_NS` for Non-secure regime.

5.17.8 Translation table walk trace

If enabled, this source traces all translation table walks initiated by the processor

Output syntax:

```
<time> <scale> <cpu> [TTW|TTU] <side> <format> <stage>:<level> <address> <data> : <result>
```

`<time>`

Timestamp (decimal value).

`<scale>`

Unit for `<time>`. `clk` indicates that the timestamp is not related to real time, but an increasing count.

`<cpu>`

Processor, or other component, that gave the instruction.

`[TTW|TTU]`
Translation table walks (reads) or translation table update (writes).

`<side>`
Data or instruction TLB.

`<format>`
VMSA or LPAE format translation table entries.

`<stage>`
Walk stage, within the range 1-2.

`<level>`
Walk level, within the range 1-3.

`<address>`
Physical address of lookup in hexadecimal.

`<data>`
Raw translation table entry in hexadecimal.

`<result>`
Parsed result. One of the following options:
`ABORTED`
The memory access caused a synchronous abort and no data was returned.
`FAULT`
The data that was returned is not valid for this stage and level.
`RESERVED`
The data that was returned is not valid for this stage and level.
`TABLE {<attr>=<value>}+`
Pointer to the next level of lookup, in LPAE format.
`BLOCK {<attr>=<value>}`
LPAE region descriptor.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

`SUPERSECTION {<attr>=<value>}`
: VMSA region descriptor.

`SECTION {<attr>=<value>}`
: VMSA region descriptor.

`PAGETABLE {<attr>=<value>}`
: Pointer to the next level of lookup, in VMSA format.

`LARGEPAGE {<attr>=<value>}`
: VMSA region descriptor.

`SMALLPAGE {<attr>=<value>}`
: VMSA region descriptor.

5.17.9 TLB trace

If enabled, this source traces TLB entries that are filled and evicted by the processor.

Output syntax:

```
<time> <scale> <cpu> [TLB|WALKCACHE] FILL <id> <size> <virtualregime>:<paddr> {<memtype>}
{<attr>=<value>}+
```

```
<time> <scale> <cpu> [TLB|WALKCACHE] EVICT <id> <size> <virtualregime>
```

`<time>`

Timestamp (decimal value).

`<scale>`

Unit for `<time>`. `clk` indicates that the timestamp is not related to real time, but an increasing count.

`<cpu>`

Processor, or other component, that gave the instruction.

`<id>`

Identifies which TLB or walk cache to trace.

`<size>`

Size of the region being mapped.

`<virtualregime>`

Virtual address and regime of the region being mapped, formatted according to the common virtual regime definition.

`<paddr>`

Physical base address of mapped region, formatted according to the common address definition.

`<memtype>`

For TLB entries, the memory type of the result. One of the following options:

`Device-[G|nG][R|nR][E|nE] {(<alias>)}`
: Device memory, where:

  `[G|nG]`
  : Gathering or nongathering.

  `[R|nR]`
  : Reordering or nonreordering.

  `[E|nE]`
  : Early write acknowledgement or not.

  `<alias>`
  : Device-nGnRnE was previously known as StronglyOrdered.

`Normal [NonShareable|Shareable] Inner=<cachetype> Outer=<cachetype>`

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Normal memory, where:

[NonShareable|Shareable]
Shareability

<cachetype>
[NonCacheable|WriteBack|WriteThrough]{NonReadAllocate}{Non}{WriteAllocate}

[NonCacheable|WriteBack|WriteThrough]
Cacheability

{NonReadAllocate}
For cacheable memory, Read allocate hint. (Read allocate is assumed if not specified.)

{Non}{WriteAllocate}
For cacheable memory, Write allocate hint.

Related references
5.17.2 TarmacTrace file format on page 5-2939

5.17.10 Event trace

If enabled, this source traces exceptions, interrupts, and exception returns. In AArch64, it also traces changes to the SPSel and to the current exception level, by generating a CoreEvent_ModeChange.

Output syntax:

`<time> <scale> {<cpu>} E <value> {<mode>} {<value1>} <number> <desc>`

<time>

Timestamp (decimal value).

<scale>

Unit for `<time>`, which gives consistency with device-specific Tarmac Trace formats.

<cpu>

Processor, or other component, that gave the instruction.

<value>

A value that is associated with the event. Format according to the common address definition.

<mode>
For mode change events only, the new mode being entered.

<value1>
Where available, the hexadecimal representation of a second value that is associated with the event.

<number>

Event number.

<desc>

Event name.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 5-20 Supported values for value, number, and desc

Number	Event description	Value
0x00000001	CoreEvent_Reset	-
0x00000002	CoreEvent_UndefinedInstr	-
0x00000003	CoreEvent_SWI	SWI number
0x00000004	CoreEvent_PrefetchAbort	-
0x00000005	CoreEvent_DataAbort	-
0x00000007	CoreEvent_IRQ	-
0x00000008	CoreEvent_FIQ	-
0x0000000E	CoreEvent_ImpDataAbort	-
0x00000019	CoreEvent_ModeChange	New mode
0x00000080	CoreEvent_CURRENT_SP0_SYNC	-
0x00000081	CoreEvent_CURRENT_SP0_IRQ	-
0x00000082	CoreEvent_CURRENT_SP0_FIQ	-
0x00000083	CoreEvent_CURRENT_SP0_ABORT	-
0x00000084	CoreEvent_CURRENT_SPx_SYNC	-
0x00000085	CoreEvent_CURRENT_SPx_IRQ	-
0x00000086	CoreEvent_CURRENT_SPx_FIQ	-
0x00000087	CoreEvent_CURRENT_SPx_ABORT	-
0x00000088	CoreEvent_LOWER_64_SYNC	-
0x00000089	CoreEvent_LOWER_64_IRQ	-
0x0000008A	CoreEvent_LOWER_64_FIQ	-
0x0000008B	CoreEvent_LOWER_64_ABORT	-
0x0000008C	CoreEvent_LOWER_32_SYNC	-
0x0000008D	CoreEvent_LOWER_32_IRQ	-
0x0000008E	CoreEvent_LOWER_32_FIQ	-
0x0000008F	CoreEvent_LOWER_32_ABORT	-

Note

The CoreEvent_CURRENT_* and CoreEvent_LOWER_* events cover all the ways in which exception entry can happen in AArch64 state. For example, CoreEvent_CURRENT_SPx_SYNC corresponds to a

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

synchronous exception taken from *Current Exception level with SP_ELx, x>0*.
CoreEvent_LOWER_64_IRQ corresponds to an IRQ or vIRQ taken from *Lower Exception level, where the implemented level immediately lower than the target level is using AArch64*.

Related references

5.17.2 TarmacTrace file format on page 5-2939

5.17.11 Processor memory access trace

If enabled, this source traces processor data accesses.

Output syntax:

```
<time> <scale> {<cpu>} M<rw><sz><attrib> <addr> <data>
```

<time>

Timestamp (decimal value).

<scale>

Unit for <time>. This element gives consistency with device-specific Tarmac Trace formats.

<cpu>

Processor, or other component, that gave the instruction.

<rw>

R
Read access.

W
Write access.

<sz>

Size of the data transfer in bytes (1, 2, 4, 8).

<attrib>

Optional access attribute:

X
Exclusive access.

T
Translated (unprivileged) access.

L
Locked access (SWP, SWPB instructions).

<addr>

Virtual address that is used to access memory. Format according to the common address definition.

<data>

Hexadecimal value of data transferred. The data padding is according to the size of the transfer. Data of 64 bits or more contains an underscore (_) separator every eight characters (32 bits).

5.17.12 Processor memory update trace

If enabled, this source traces memory update accesses caused by atomic operations.

Output syntax:

```
<time> <scale> {<cpu>} MU<sz>_<atomic_op> <addr> <data>
```


Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

<time>

Timestamp (decimal value).

<scale>

Unit for <time>. This element gives consistency with device-specific Tarmac Trace formats.

<cpu>

Processor, or other component, that gave the instruction.

<sz>

Size of the data transfer in bytes (1, 2, 4, 8, 16).

<atomic_op>

Atomic operation performed on this memory address:

ADD
Atomic add operation.

BIC
Atomic bit clear operation.

CAS
Atomic compare and swap operation.

EOR
Atomic exclusive or operation.

ORR
Atomic bit set operation.

SMAX
Atomic signed max operation.

SMIN
Atomic signed min operation.

SWP
Atomic swap operation.

UMAX
Atomic unsigned max operation.

UMIN
Atomic unsigned min operation.

<addr>

Physical address that is used to access memory. Format according to the common address definition.

<data>

Hexadecimal value of the data transferred. The data padding is according to the size of the transfer. Data of 64 bits or more contains an underscore (_) separator every eight characters (32 bits).

5.17.13 Memory bus trace

If enabled, this source traces transactions that are initiated through the memory bus master port of the processor. These accesses use physical addresses.

Output syntax:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

`<time> <scale> {<cpu>} B<rw><sz><fd><lk><p><s> I<wrcbs> O<wrcbs> <master_id> <addr> <data>`

`<time>`

Timestamp (decimal value).

`<scale>`

Unit for `<time>`. This element gives consistency with device-specific Tarmac Trace formats.

`<cpu>`

Processor, or other component, that gave the instruction.

`<rw>`

`R`
Read access.

`W`
Write access.

`<sz>`

Size of the data transfer in bytes.

`<fd>`

`I`
Opcode fetch.

`D`
Data load/store or an MMU access.

`<lk>`

`L`
Locked access.

`X`
Exclusive access.

_, underscore
Normal access.

`<p>`

`P`
Privileged access.

_, underscore
Normal access.

`<s>`

`S`
Secure access.

`N`
Non-secure access.

`I<wrcbs>`

Inner cache attributes. See `O<wrcbs>`.

`O<wrcbs>`

Outer cache attributes:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

<w>

W
Allocate on write.

_, underscore
No allocate on write.

<r>

R
Allocate on read.

_, underscore
No allocate on read.

<c>

C
Cacheable access.

_, underscore
Non-cacheable access.

<b>

B
Bufferable access.

_, underscore
Non-bufferable access.

<s>

S
Shareability access.

_, underscore
Non-shareability access.

<master_id>

Master ID of the transaction.

<addr>

Physical address that is used to access memory, in hexadecimal format.

<data>

Hexadecimal value of data transferred. The data padding is according to the size of the transfer. Byte ordering is from lowest to highest byte. This ordering means that for accesses in little endian mode, the data occurs mirrored compared to the register/memory access records.

5.17.14 Tarmac Trace output example

This is an example of a trace that was produced by the Tarmac Trace plug-in, showing instruction, register, event, and memory access traces.

```
1939 clk cpu0 IT (1915) 0001129c:00001521129c d51bd061 O EL3h_s : MSR TPIDRRO_EL0,x1
1939 clk cpu0 R TPIDRRO_EL0 00000000:00000000
1940 clk cpu0 IT (1916) 000112a0:0000152112a0 d001c100 O EL3h_s : ADRP x0,{pc}+0x3822000 ;
0x38332a0
1940 clk cpu0 R X0 0000000003833000
1941 clk cpu0 IT (1917) 000112a4:0000152112a4 91024000 O EL3h_s : ADD x0,x0,#0x90
1941 clk cpu0 R X0 0000000003833090
1942 clk cpu0 IT (1918) 000112a8:0000152112a8 d2810002 O EL3h_s : MOV x2,#0x800
1942 clk cpu0 R X2 0000000000000800
1943 clk cpu0 IT (1919) 000112ac:0000152112ac 91000421 O EL3h_s : ADD x1,x1,#1
1943 clk cpu0 R X1 0000000000000001
1944 clk cpu0 IT (1920) 000112b0:0000152112b0 9b017c42 O EL3h_s : MUL x2,x2,x1
```


Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

```
1944 clk cpu0 R X2 0000000000000800
1945 clk cpu0 IT (1921) 000112b4:0000152112b4 8b020000 O EL3h_s : ADD x0,x0,x2
1945 clk cpu0 R X0 0000000003833890
1946 clk cpu0 IT (1922) 000112b8:0000152112b8 9100001f O EL3h_s : MOV sp,x0
1946 clk cpu0 R SP_EL3 0000000003833890
1947 clk cpu0 IT (1923) 000112bc:0000152112bc 10001420 O EL3h_s : ADR x0,{pc}+0x284 ; 0x11540
1947 clk cpu0 R X0 0000000000011540
1947 clk cpu0 CACHE cpu.cpu0.l1icache LINE 0096 ALLOC 0x0000152112c0
1947 clk cpu0 CACHE Validation_ARMAEMv8AMPCT.cpu.l2_cache LINE 04b0 ALLOC 0x0000152112c0
1948 clk cpu0 IT (1924) 000112c0:0000152112c0 f9400000 O EL3h_s : LDR x0,[x0,#0]
1948 clk cpu0 MR8 00011540:000015211540 00000000_13000000
1948 clk cpu0 R X0 0000000013000000
1948 clk cpu0 CACHE cpu.cpu0.l1dcache LINE 00aa ALLOC 0x000015211540
1948 clk cpu0 CACHE Validation_ARMAEMv8AMPCT.cpu.l2_cache LINE 0550 ALLOC 0x000015211540
1949 clk cpu0 IT (1925) 000112c4:0000152112c4 d0000081 O EL3h_s : ADRP x1,{pc}+0x12000 ;
0x232c4
1949 clk cpu0 R X1 0000000000023000
1950 clk cpu0 IT (1926) 000112c8:0000152112c8 91017021 O EL3h_s : ADD x1,x1,#0x5c
1950 clk cpu0 R X1 000000000002305C
1951 clk cpu0 IT (1927) 000112cc:0000152112cc d63f0020 O EL3h_s : BLR x1
1951 clk cpu0 R X30 00000000000112D0
1951 clk cpu0 TTW ITLB LPAE 1:3 000016390010 00000000152204c3 : BLOCK ATTRIDX=0 NS=0 AP=3
SH=0 AF=1 nG=0 16E=0 PXN=0 XN=0 ADDR=0x0000000015220000
1951 clk cpu0 TLB FILL cpu.cpu0.ITLB 64K 0x00020000, nG asid=0:0x0015220000 Normal
NonShareable Inner=WriteBackWriteAllocate Outer=WriteBackWriteAllocate xn=0 pxn=0
ContiguousHint =0
1951 clk cpu0 TLB FILL cpu.cpu0.S1TLB 64K 0x00020000, nG asid=0:0x0015220000 Normal
NonShareable Inner=WriteBackWriteAllocate Outer=WriteBackWriteAllocate xn=0 pxn=0
ContiguousHint =0
1951 clk cpu0 CACHE cpu.cpu0.l1icache LINE 0182 ALLOC 0x000015223040
1951 clk cpu0 CACHE Validation_ARMAEMv8AMPCT.cpu.l2_cache LINE 0c10 ALLOC 0x000015223040
1952 clk cpu0 IT (1928) 0002305c:00001522305c f0030f48 O EL3h_s : ADRP x8,{pc}+0x61eb000 ;
0x620e05c
1952 clk cpu0 R X8 000000000620E000
1953 clk cpu0 IT (1929) 00023060:000015223060 f9000100 O EL3h_s : STR x0,[x8,#0]
1953 clk cpu0 TTW DTLB LPAE 1:3 000016393100 0000000016000463 : BLOCK ATTRIDX=0 NS=1 AP=1
SH=0 AF=1 nG=0 16E=0 PXN=0 XN=0 ADDR=0x0000000016000000
1953 clk cpu0 MW8 0620e000:00001600e000_NS 00000000_13000000
1953 clk cpu0 TLB FILL cpu.cpu0.DTLB 64K 0x06200000, nG asid=0:0x0016000000_NS Normal
NonShareable Inner=WriteBackWriteAllocate Outer=WriteBackWriteAllocate xn=0 pxn=0
ContiguousHint =0
1953 clk cpu0 TLB FILL cpu.cpu0.S1TLB 64K 0x06200000, nG asid=0:0x0016000000_NS Normal
NonShareable Inner=WriteBackWriteAllocate Outer=WriteBackWriteAllocate xn=0 pxn=0
ContiguousHint =0
1953 clk cpu0 CACHE cpu.cpu0.l1dcache LINE 0188 ALLOC 0x000016393100
1953 clk cpu0 CACHE Validation_ARMAEMv8AMPCT.cpu.l2_cache LINE 0c40 ALLOC 0x000016393100
1954 clk cpu0 IT (1930) 00023064:000015223064 17fffff5 O EL3h_s : B {pc}-0x2c ; 0x23038
1954 clk cpu0 CACHE cpu.cpu0.l1icache LINE 0180 ALLOC 0x000015223000
```


Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Chapter 6
Fast Models examples

The following top-level example directories are installed in `$PVLIB_HOME/examples`.

Table 6-1 Fast Models examples directories

Directory name	Description
`CADI`	Example C++ applications that demonstrate how to use the CADI debug API. **Note** CADI is deprecated and is being replaced by the Iris debug API.
`LISA`	LISA+ source code and project files for FVPs.
`LISAPlus`	Example LISA+ components that show how to capture and generate MTI trace, remap `PVBus` transactions, and handle burst transactions.
`MTI`	MTI plug-in examples that show how to extract and use trace information from models. The following examples are also available as pre-built libraries under `$PVLIB_HOME/plugins`: • GenericCounter. • GenericTrace. • ListTraceSources.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 6-1 Fast Models examples directories (continued)

Directory name	Description
`python`	Python debug script examples that show how to use the `fm.debug` Python module for common debugging tasks.
`SystemCExport`	• Source code and makefiles for EVS platform examples and SVPs. • LISA+ source for bridges and EVS components. • Header files required for exporting LISA+ protocols to SystemC.

Note

On Microsoft Windows, the Fast Models installer creates a copy of the examples in `%USERPROFILE%\ARM\FastModelsPortfolio_%FM-VERSION%\examples\`. This copy allows you to save configuration changes to the examples without needing Administrator permissions.

It contains the following sections:

- *6.1 CADI examples* on page 6-2955.
- *6.2 LISA examples* on page 6-2956.
- *6.3 LISAPlus examples* on page 6-2957.
- *6.4 MTI examples* on page 6-2958.
- *6.5 Python examples* on page 6-2959.
- *6.6 SystemCExport examples* on page 6-2960.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

6.1 CADI examples

Example CADI clients that demonstrate how to use the CADI API to perform common debugger operations. These include instantiating a new simulation, connecting to a running simulation, accessing memory and registers, setting breakpoints, and registering callbacks for asynchronous feedback from the simulation.

Each of these examples has a `readme.txt` that describes the example, how to run it, and in some cases, gives the expected output. The following CADI examples are provided:

Table 6-2 CADI examples

Example	Description
`breakpoints`	Sets a code breakpoint at a specific address in the application that is running on the model.
`cache_dump`	Displays information about the caches in the running simulation.
`cadi_lib`	Implements a shared library that contains a CADI simulation. You can load the library into Model Shell and connect a CADI-compliant debugger to it.
`cadi_server`	Implements an executable that contains a CADI simulation. It loads the plug-in library `CADIIPCRemoteConnection.so`, which enables you to connect a CADI-compliant debugger to it. You can run the `connecting_running_target` CADI example to mimic this connection.
`connecting_instantiating_model`	Instantiates a simulation that is a shared library, prints a list of all targets in the simulation, and connects to one of them.
`connecting_remote_target`	Connects to a remote simulation that has been configured to allow remote connections. The IP address and listening port are passed as parameters to the CADI client.
`connecting_running_target`	Connects to a running CADI target.
`disassembly`	Prints the disassembly of the instruction at the PC address in the application that is running on the model.
`memory`	Displays memory space information. Reads and displays some values from memory in the application that is running on the model.
`mti`	Loads the TarmacTrace MTI plug-in into the application that is running on the model and displays some tarmac trace output.
`registers`	Reads and displays register information for a CPU target in the model and reads and writes a specific register.
`tlb_dump`	Connects to a model and displays the contents of the TLBs.

Related information

Component Architecture Debug Interface Developer Guide

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

6.2 LISA examples

LISA+ source and project files for FVPs.

Note

The LISA platform examples are built as ISIMs. For more information about building an ISIM, see *Building an ISIM* in the Fast Models User Guide.

The following LISA examples are provided:

Table 6-3 LISA examples

Example	Description
`BusComponents`	Example LISA+ components that demonstrate different ways of using the `PVBus` interface.
`Common`	FVP-specific LISA+ components that are common to different types of FVPs.
`CSS`	Source and project files for SGI_575 and SGM_775 Reference Data FVPs.
`FVP_Base`	Source and project files for Base Platform FVPs. For information about the Base Platform, see *Chapter 7 Base Platform* on page 7-2961.
`FVP_BaseR`	Source and project files for BaseR Platform FVPs.
`FVP_Base_RevC`	Source and project files for Base Platform RevC FVPs. For information about the Base Platform RevC, see *7.2 Base Platform RevC* on page 7-2963.
`FVP_Coproc_Demo`	Example implementation of the `Coprocessor` interface. Registers the coprocessor with a ARMCortexM33CT or ARMAEMv8MCT model. For more information, see *2.6.1 CoprocBusProtocol protocol* on page 2-81.
`FVP_MPS2`	Source and project files for MPS2 FVPs. For information about the MPS2 platforms, see *Chapter 8 Microcontroller Prototyping System 2* on page 8-2988.
`FVP_VE`	Source and project files for VE FVPs. For information about the VE platform, see *Chapter 9 Versatile Express Model* on page 9-2999.
`VP_PChannel`	Shows how to create power controllers to control the power state of the cores and cluster, using the `PChannel` protocol. For information about `PChannel`, see *2.5.1 PChannel protocol* on page 2-80.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

6.3 LISAPlus examples

Source code and System Canvas project files for some example LISA+ components.

Table 6-4 LISAPlus examples

Example	Description
`CapturingTraceFromLISA`	Instantiates and uses an MTI plug-in from a LISA+ component. Uses the SimpleTrace plug-in as an example.
`GeneratingTraceFromLISA`	Generates MTI trace information from a LISA+ component.
`PVBusBursts`	Uses a `PVBusMaster` to generate burst read transactions, which are handled by a `PVBusSlave`.
`RemappingWithPVBusMapper`	Uses the `PVBusMapper` component to remap transactions based on their attributes.

Related references

3.2.10 PVBusMapper on page 3-108

3.2.11 PVBusMaster on page 3-108

3.2.14 PVBusSlave on page 3-109

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

6.4 MTI examples

Example MTI plug-ins that show how to use MTI to extract and use trace information from models.

The following MTI examples are provided:

Table 6-5 MTI examples

Example	Description
`CallTrace`	Displays a function call sequence by tracing the `PC` field of `INST` trace sources, then compares the output with values in a symbol table. See the `readme` for more information.
`CountingCacheStats`	Registers counters for cache-related trace sources, for example `CACHE_READ_HIT`. Prints the cache stats before terminating.
`DCCTrace`	Prints the value of `DBGDTRxX_EL0` when data is written. Updates the `TXFull` bit in MDSCR to indicate the data was read. See the `readme` for more information.
`GenericCounter`	Registers a counter for trace sources. Prints the counter value for each `INST` trace source before terminating. This example is also available as a pre-built library, see *5.9 GenericCounter* on page 5-2921.
`GenericTrace`	A flexible plug-in that traces one or more trace sources specified by the user. Prints the trace to a text file or to stdout. This example is also available as a pre-built library, see *5.10 GenericTrace* on page 5-2922.
`ListTraceSources`	Displays either the trace sources provided by all trace components in the model, or just the trace components, to a text file or to stdout, without running the simulation. For more information, see `readme.txt`. This example is also available as a pre-built library, see *5.11 ListTraceSources* on page 5-2924.
`RunTimeParameterTest`	Uses MTI to set runtime parameters.
`SimpleTrace`	Simple trace plug-in that prints a trace of the PC.
`SoftwareTrigger`	Traces `SEMIHOSTING_PRECALL` trace events, intercepts semihosting calls, and prints out register information. For more information, see the `readme`.
`TraceOnBreak`	Similar to the `SimpleTrace` example, but prints the PC value only when a breakpoint is hit.

Related information

Fast Models Model Trace Interface Reference Manual

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

6.5 Python examples

Python debug scripting examples.

These examples show how to use the `fm.debug` Python module to connect to targets, read the contents of caches, read/write memory and registers, set breakpoints, and perform run control. Some of these scripts use images that are supplied as assembly language source files.

Table 6-6 Python examples

Example	Description
`checkpointing.py`	Saves or restores the state of a library model, or a specific component in it, using either the default checkpoint delegate, or an example of a custom delegate.
`DemoBreakpoints.py`	Enables exception breakpoints and sets program and memory breakpoints in an application running on a library model. Prints the breakpoints that were hit.
`DemoCaches.py`	Prints the contents of `PVCache` CADI targets that are selected by the user.
`DemoDumpAllRegisters.py`	Prints register names and values. You can specify a list of registers and targets, or if not, it prints all registers for all CPU targets.
`DemoMemory.py`	Writes to and reads from target memory, as bytes and words, to show the effect of toggling the endianness of the processor.
`DemoParameters.py`	Gets and sets instantiation time and run-time model parameters.
`DemoRegisters.py`	Reads and writes registers. Shows different ways of specifying register names.
`DemoSemihosting.py`	Binds semihosting stdin and stdout calls to the semihosting stdin and stdout of the target.
`DemoStopOnStringOut.py`	Stops the model when a string is entered in the UART terminal.
`DemoTLB.py`	Prints the contents of the `TlbCadi` CADI targets selected by the user.
`DemoTraceOnBpt.py`	Starts tracing a model from the address at which a breakpoint is hit. The GenericTrace plug-in must have been loaded on the model that it connects to. For information about GenericTrace, see *5.10 GenericTrace* on page 5-2922.
`LoadModel.py`	Loads a library model that is specified as a parameter.

Related information

Python Debug Scripting for Fast Models Reference Manual

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

6.6 SystemCExport examples

Components and platform models that are created by exporting LISA+ components or platforms to SystemC. Also, bridge components for converting transactions between LISA+ protocols and SystemC.

Table 6-7 SystemCExport examples

Directory	Description
`Bridges`	LISA+ source for bridge components. For more information, see *3.1 Bridge components* on page 3-88.
`Common`	Source files and makefile rules that are common to the EVS and SVP examples.
`Common/Protocols`	Header files that are required for the export of LISA+ protocols to SystemC.
`EVS_Components`	LISA+ files and project files for EVS (Exported Virtual Subsystem) components. These are LISA+ components with a SystemC wrapper and bridges that allow them to be used in an SVP. Build the component to generate the header file and libraries required to include it in a SystemC simulation.
`EVS_Platforms`	LISA+ source and makefiles for EVS platform examples. An EVS platform is a LISA+ platform that has been exported as a SystemC object to allow it to be integrated into a SystemC simulation. The EVS platform examples are minimal platforms that are designed for a specific use case, for example running the Dhrystone benchmark application or booting Linux. The Dhrystone images `dhrystone_v8.axf` and `dhrystone.axf` for Armv7, are provided in the Fast Models Third Party IP package. They are loaded from `$PVLIB_HOME/images/`. For more information about building EVS platforms, see *Building Fast Models* in the Fast Models User Guide.
`SVP_Platforms`	SVPs (SystemC Virtual Platforms) are platform models in which each component or subsystem has been individually exported to SystemC using the Fast Models Multiple Instantiation (MI) feature. For more information, see *Building an SVP* in the Fast Models User Guide. SVP platforms can be modified by replacing EVS components with other Fast Models EVSs, or with native SystemC components. There are three subdirectories: **SVP_Base** Armv8 Base Platform SVPs. **SVP_bigLITTLE** Armv7 big.LITTLE™ SVP, containing Cortex-A15 and Cortex-A7 clusters. **SVP_LinuxBoot** Minimal Armv7 platform that is suitable for booting Linux, containing a Cortex-A15 cluster.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Chapter 7
Base Platform

This chapter describes the Base Platform system model.

It contains the following sections:

- *7.1 Base - about* on page 7-2962.
- *7.2 Base Platform RevC* on page 7-2963.
- *7.3 Base - memory* on page 7-2967.
- *7.4 Base - interrupt assignments* on page 7-2971.
- *7.5 Base - clocks* on page 7-2974.
- *7.6 Base - parameters* on page 7-2975.
- *7.7 Base - components* on page 7-2976.
- *7.8 Base - differences between the AEMv8-A FVP and core FVPs* on page 7-2984.
- *7.9 Base - VE compatibility* on page 7-2985.
- *7.10 Base - unsupported VE features* on page 7-2987.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

7.1 Base - about

The Base Platform system model allows early development, distribution, and demonstration of software deliverables. A range of Base FVPs are supplied as standalone products and as examples in Fast Models.

For a list of the Base and BaseR Platform FVPs and the instances in them, see *Base Platform FVPs* and *BaseR Platform FVPs* in the FVP Reference Guide.

See `$PVLIB_HOME/examples/LISA/FVP_Base/` for LISA+ source and project files for the Base Platform FVP examples.

The standard peripheral set enables software development and porting. The platform is an evolution of the VE *Fixed Virtual Platforms* (FVPs), based on the Arm Versatile Express (VE) hardware development platform.

The Base Platform system model provides:

- Two configurable clusters of up to four core models that implement:
  - AArch64 at all exception levels.
  - Configurable AArch32 support at all exception levels.
  - Configurable support for little and big endian at all exception levels.
  - Generic timers.
  - Self-hosted debug.
  - CADI debug.
  - GICv3 memory-mapped processor interfaces and distributor.
- Peripherals for multimedia or networking environments.
- Four PL011 UARTs.
- A CoreLink™ CCI-400 Cache Coherent Interconnect.
- Architectural GICv3 model.
- High Definition LCD Display Controller, 1920×1080 resolution at 60fps, with single I2S and four stereo channels.
- 64MB NOR flash and board peripherals.
- CoreLink TZC-400 TrustZone Address Space Controller.

Related references

6.2 LISA examples on page 6-2956

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

7.2 Base Platform RevC

Base Platform RevC is a configuration of the Base Platform that includes a PCI subsystem.

LISA+ source and project files for Base Platform RevC FVPs are located in `$PVLIB_HOME/examples/LISA/FVP_Base_RevC/`.

This section contains the following subsections:

- *7.2.1 Base Platform RevC changes* on page 7-2963.
- *7.2.2 BasePlatformPCIRevC component* on page 7-2964.

7.2.1 Base Platform RevC changes

The Base Platform RevC includes the following additions to the previous Base Platform version.

- A PCIe config region and two PCIe memory regions. See the memory map for details.
- A CoreLink CCI-550 Cache Coherent Interconnect.
- An AHCI controller model including a SATA disk model.
- An SMMUv3 AEM. This is an architectural model that implements the SMMUv3.0 and SMMUv3.1 architectures. These architectures are for I/O virtualization of devices. The SMMU is placed so that accesses to memory by PCI devices acting as bus masters are affected by it.

  The SMMU has the following features:

  — Memory that is mapped to the range `0x2B400000-0x2B4FFFFF`.
  — Interrupts with IRQ IDs in the range 103-111.
  — The event output pin of the SMMU is passed to the clusters.
  — The downstream ports of the SMMU attach to the coherent bus infrastructure and so are coherent with the core clusters. All cores and the SMMU are in the same shareability domain. There is no distinction between the inner and outer shareability domains.
  — The parameters of the SMMU determine its capabilities and have default values which can be overwritten if necessary.
  — The SMMU is configured to only accept 16-bit StreamIDs and there is a 1:1 correspondence between RequestorID and StreamID.
  — By default, the SMMU uses DeviceID `0x10000` to identify itself to the GIC (`pci.pci_smmuv3.mmu.smmu_msi_device_id`).

    The parameter `gic_distributor.ITS-device-bits` is set to 17 by default to support the 17-bit DeviceIDs.

  The SMMU has the following limitations:
  — It does not support RAS.
  — The PMU has limited functionality. Only a subset of the architecturally mandatory events are supported, as indicated by the `SMMU_PMCG_CEID0` fields. The PMU is intended for demonstration purposes only and for driver development.
- An SMMUv3TestEngine component with a control region at `0x002bfe0000-0x002bfFFFFF`. This component is a traffic generator that acts as a (secure) device upstream of the SMMUv3.
- Two PCIe virtio devices are above the SMMU. By default they are configured to be device 0 and 2 on bus 0.
- The PCIe devices use a DeviceID that is the same as their RequestorID (BDF).
- Legacy PCI interrupts:
  — Each PCI device is hardwired to use INTA, with a value of 1 in the `interrupt_pin` register. This is required by the PCI specification for single-function devices.
  — The interrupts in the PCI host bridge are mapped according to section 2.2.6 of the PCI Local Bus Specification Revision 3.0, using the following formula, where the values for DeviceInterrupt are INTA = 0, INTB = 1, INTC = 2, INTD = 3:

  ```
  BridgeInterrupt = (Device + DeviceInterrupt) % 4
  ```

  This formula produces the following mappings:

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

BridgeInterrupt ID	DeviceInterrupt
200	INTA
201	INTB
202	INTC
203	INTD
207	SERR

- The model optionally implements MSI-X, depending on whether a parameter is set. If this parameter is set, an MSI-X capability is advertised as a PCI capability.

  Note

  The virtio specification is not fully compliant with the PCI specification and the virtio block device cannot be used in a pure-polling mode where MSI-X is always masked and only polling the pending bit array is used.

  The MSIs produced by the models, when directed to the GIC, have their payload rewritten to carry the DeviceID of the originating device to the GIC.
- The processor models implement architecture version v8.0, which does not support the Statistical Profiling Extension (SPE). To include SPE, add parameter `cluster0.has_arm_v8-3=1`, or similar, to the command line.

Related references

7.3.2 Base Platform memory map on page 7-2967
7.4 Base - interrupt assignments on page 7-2971

7.2.2 BasePlatformPCIRevC component

This component is an integrated PCIe subsystem which forms part of the Base Platform RevC. It incorporates an SMMUv3, a PCIe, an AHCI controller, and two PCI devices which wrap a pair of virtio PCI block devices. This model is written in LISA+.

Note

- You can include this component in a platform model, but Arm does not support using its subcomponents to create your own PCIe platform.
- The PCIe is not an implementation of any specific IP, but a functional, and limited, implementation of the PCIe standard.

`BasePlatformPCIRevC` is composed of the following model components:

`pci.pvbus2pci`

The bridge from the Programmer's View bus to the PCI bus.

`pci.pcidevice<n>`

A wrapper around the underlying virtio block device. There are two block devices in the system, 0 and 1.

`pci.pcivirtioblockdevice<n>`

The instances of the virtio block device component.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

pci.ahci_pci.ahci

An AHCI_SATA component with the following features:

- Each AHCI controller supports up to 32 simulated SATA disks. The configuration parameter `image_path` is a comma-separated list of one or more disk images.
- Supports 64-bit addresses.
- Supports plain, linear disk images, but also works with sparse files.

Some interesting options are:

- If the following options are set to non-zero values, they print messages about the operation of the bridge. The higher the value, the more verbose the component is:

```
pci.pvbus2pci.diagnostics=0x0    # (int) default = '0x0': Diagnostics level: [0x0..0x4]
pci.pcidevice<N>.diagnostics=0x0 # (int) default = '0x0': Diagnostics level: [0x0..0x4]
```

- Each PCI device uses three BARs; one for config space, one for the MSI-X table structure and one for the MSI-X Pending Bit Array. Each of these can be configured to be 32 bits or 64 bits wide.

  The Bus and Device number can be configured for each PCI device. If the device advertises MSI-X, support can be configured.

```
pci.pcidevice<N>.bus=0x0        # (int ) default = '0x0' : Bus number for this device : [0x0..0xFF]
pci.pcidevice<N>.device=0x0     # (int ) default = '0x0' : Device number on this bus : [0x0..0x1F]
pci.pcidevice<N>.bar0_64bit=0   # (bool) default = '0' : If BAR 0 is 64 bits wide, if region size is nonzero
pci.pcidevice<N>.msix_support=0 # (bool) default = '0' : Enable device support for MSI-X
pci.pcidevice<N>.bar2_64bit=0   # (bool) default = '0' : If BAR 2 is 64 bits wide, if region size is nonzero
pci.pcidevice<N>.bar4_64bit=0   # (bool) default = '0' : If BAR 4 is 64 bits wide, if region size is nonzero
```

- The following option configures the image file that the virtio block device exposes:

```
pci.pcivirtioblockdevice<N>.image_path="" # (string) default = '' : image file path
```

- There are two `PVBusLogger`s in the `pvbus2pci` component. One is in front of the Configuration space and one is in front of the Device space:

```
FVP_Base_AEMv8A_AEMv8A-PCI.pci.pvbus2pci.devicelogger
FVP_Base_AEMv8A_AEMv8A-PCI.pci.pvbus2pci.cfglogger
```

- There is one `PVBusLogger` in the `pcidevice` component. This reports on DMA accesses by the PCI device:

```
FVP_Base_AEMv8A_AEMv8A-PCI.pci.pcidevice0.dmalogger
```

- There is a `PVBusLogger` downstream of the SMMU. This reports on the transactions after they have been transformed by the SMMU:

```
FVP_Base_AEMv8A_AEMv8A_PCI.pci.smmulogger
```

  For example, you can see all accesses to device space by adding the following options to the command line:

```
--plugin GenericTrace.so
-C TRACE.GenericTrace.trace-sources="FVP_Base_AEMv8A_AEMv8A-PCI.pci.pvbus2pci.devicelogger.*"
```

- To supply the AHCI controller with one or more SATA disk images, use the `image_path` parameter. For example:

```
-C pci.ahci_pci.ahci.image_path=disk1tb.img,disk8tb.img
```

Table 7-1 BasePlatformPCIRevC ports

Name	Protocol	Type	Description
`pvbus_address_map_s`	PVBus	Slave	Input port to service transactions based on the PVBus protocol.
`pvbus_address_map_m`	PVBus	Master	Output port to send out PVBus transactions that are not handled by this component.
`system_reset`	Signal	Slave	Input port to handle reset signals. It is used to reset the internal state of this component.
`sev_out`	Signal	Peer	Port to send out a notification of the occurrence of an event as sg::Signal to a peer.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 7-1 BasePlatformPCIRevC ports (continued)

Name	Protocol	Type	Description
`interrupts[224]`	Signal	Master	Array of output ports of type sg::Signal to send out interrupts generated by this component.
`pvbus_pci_dma_m`	PVBus	Master	Output port to send out any DMA (of PVBus protocol) accesses originating from this component.
`clk_in`	ClockSignal	Slave	Input port to connect to a ClockSignal provider.

Table 7-2 BasePlatformPCIRevC parameters

Name	Type	Allowed values	Default value	Description
`ITS0-base`	`uint64_t`	`0-0xFFFFFFFFFFFFFFFF`	`0x2f020000`	The ITS0 Base address.
`pci_smmuv3.mmu.SMMU_IDR1`	`uint32_t`	`0-0xFFFFFFFF`	`0xe739d10`	SMMU_IDR1.
`pci_smmuv3.mmu.smmu_msi_device_id`	`uint32_t`	`0-0xFFFFFFFF`	`0x10000`	When appropriately enabled, assume that MSIs that are generated by the SMMU are presented to the GIC with this DeviceID. See parameter `msi_attribute_transform` and `enable_device_id_checks`.

Related references

3.9.64 SMMUv3AEM on page 3-836
3.7.34 VirtioBlockDevice on page 3-637

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

7.3 Base - memory

This section describes the memory of the Base Platform.

This section contains the following subsections:

- *7.3.1 Base - secure memory* on page 7-2967.
- *7.3.2 Base Platform memory map* on page 7-2967.
- *7.3.3 BaseR Platform memory map* on page 7-2969.
- *7.3.4 Base - DRAM* on page 7-2970.

7.3.1 Base - secure memory

Enable security checking on memory transactions by the TZC-400 by using the `bp.secure_memory` parameter.

Table 7-3 Secure and Non-secure access permissions

Security	`bp.secure_memory=false`	`bp.secure_memory=true`
S	Secure and Non-secure access permitted.	Secure access is permitted, Non-secure access aborts.
S/NS	Secure and Non-secure access permitted.	Secure and Non-secure access permitted.
P	Secure and Non-secure access permitted.	Access conditions are programmable by the TZC-400.

Note

The default state of the TZC-400 is to abort all accesses, even from Secure state.

Table 7-4 NSAIDs and filters that masters present to the TZC-400

Component	NSAID[u]	Filter
Cluster 0	9	0
Cluster 1	9	0
VirtIO	8	0
HDLCD0	2	2
CLCD	1	2

7.3.2 Base Platform memory map

The Base Platform memory map is based on the Versatile Express RS2 memory map with extensions.

Table 7-5 Base Platform memory map

Peripheral	Start address	Size	End address	Security
Trusted Boot ROM, secure flash, IntelStrataFlashJ3	`0x00_0000_0000`	64MB	`0x00_03FF_FFFF`	S
Trusted SRAM	`0x00_0400_0000`	256KB	`0x00_0403_FFFF`	S
Trusted DRAM	`0x00_0600_0000`	32MB	`0x00_07FF_FFFF`	S
NOR flash, flash0, IntelStrataFlashJ3	`0x00_0800_0000`	64MB	`0x00_0BFF_FFFF`	S/NS
NOR flash, flash1, IntelStrataFlashJ3	`0x00_0C00_0000`	64MB	`0x00_0FFF_FFFF`	S/NS

[u] *Non-Secure Access IDentity.*

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 7-5 Base Platform memory map (continued)

Peripheral	Start address	Size	End address	Security
CS1-Pseudostatic RAM, PSRAM, on the motherboard.	0x00_1400_0000	64MB	0x00_17FF_FFFF	S/NS
VRAM	0x00_1800_0000	32MB	0x00_19FF_FFFF	S/NS
Ethernet, SMSC 91C111	0x00_1A00_0000	16MB	0x00_1AFF_FFFF	S/NS
USB, unimplemented	0x00_1B00_0000	16MB	0x00_1BFF_FFFF	S/NS
VE System Registers	0x00_1C01_0000	64KB	0x00_1C01_FFFF	S/NS
System Controller, SP810	0x00_1C02_0000	64KB	0x00_1C02_FFFF	S/NS
AACI, PL041	0x00_1C04_0000	64KB	0x00_1C04_FFFF	S/NS
MCI, PL180	0x00_1C05_0000	64KB	0x00_1C05_FFFF	S/NS
KMI - Keyboard, PL050	0x00_1C06_0000	64KB	0x00_1C06_FFFF	S/NS
KMI - Mouse, PL050	0x00_1C07_0000	64KB	0x00_1C07_FFFF	S/NS
UART0, PL011	0x00_1C09_0000	64KB	0x00_1C09_FFFF	S/NS
UART1, PL011	0x00_1C0A_0000	64KB	0x00_1C0A_FFFF	S/NS
UART2, PL011	0x00_1C0B_0000	64KB	0x00_1C0B_FFFF	S/NS
UART3, PL011	0x00_1C0C_0000	64KB	0x00_1C0C_FFFF	S/NS
Watchdog, SP805	0x00_1C0F_0000	64KB	0x00_1C0F_FFFF	S/NS
Base Platform Power Controller	0x00_1C10_0000	64KB	0x00_1C10_FFFF	S/NS
Dual-Timer 0, SP804	0x00_1C11_0000	64KB	0x00_1C11_FFFF	S/NS
Dual-Timer 1, SP804	0x00_1C12_0000	64KB	0x00_1C12_FFFF	S/NS
Virtio block device	0x00_1C13_0000	64KB	0x00_1C13_FFFF	S/NS
Virtio Plan 9 device	0x00_1C14_0000	64KB	0x00_1C14_FFFF	S/NS
Virtio net device	0x00_1C15_0000	64KB	0x00_1C15_FFFF	S/NS
Real-time Clock, PL031	0x00_1C17_0000	64KB	0x00_1C17_FFFF	S/NS
CF Card, unimplemented	0x00_1C1A_0000	64KB	0x00_1C1A_FFFF	S/NS
Color LCD Controller, PL111	0x00_1C1F_0000	64KB	0x00_1C1F_FFFF	S/NS
Non-trusted ROM, nontrustedrom	0x00_1F00_0000	4KB	0x00_1F00_0FFF	S/NS
CoreSight and peripherals	0x00_2000_0000	128MB	0x00_27FF_FFFF	S/NS
REFCLK CNTControl, Generic Timer	0x00_2A43_0000	4KB	0x00_2A43_0FFF	S
EL2 Generic Watchdog Control	0x00_2A44_0000	4KB	0x00_2A44_0FFF	S/NS
EL2 Generic Watchdog Refresh	0x00_2A45_0000	4KB	0x00_2A45_0FFF	S/NS
Trusted Watchdog, SP805	0x00_2A49_0000	64KB	0x00_2A49_FFFF	S
TrustZone Address Space Controller, TZC-400	0x00_2A4A_0000	4KB	0x00_2A4A_0FFF	S
REFCLK CNTRead, Generic Timer	0x00_2A80_0000	4KB	0x00_2A80_0FFF	S/NS
AP_REFCLK CNTCTL, Generic Timer	0x00_2A81_0000	4KB	0x00_2A81_0FFF	S/NS
AP_REFCLK CNTBase0, Generic Timer	0x00_2A82_0000	4KB	0x00_2A82_0FFF	S
AP_REFCLK CNTBase1, Generic Timer	0x00_2A83_0000	4KB	0x00_2A83_0FFF	S/NS

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 7-5 Base Platform memory map (continued)

Peripheral	Start address	Size	End address	Security
DMC-400 CFG, unimplemented	0x00_2B0A_0000	64KB	0x00_2B0A_FFFF	S/NS
SMMUv3 AEM[v]	0x00_2B40_0000	1MB	0x00_2B4F_FFFF	S/NS
GIC Physical CPU interface, GICC[w]	0x00_2C00_0000	8KB	0x00_2C00_1FFF	S/NS
GIC Virtual Interface Control, GICH[w]	0x00_2C01_0000	8KB	0x00_2C01_1FFF	S/NS
GIC Virtual CPU Interface, GICV[w]	0x00_2C02_F000	8KB	0x00_2C03_0FFF	S/NS
CCI-400	0x00_2C09_0000	64KB	0x00_2C09_FFFF	S/NS
Mali-G76[x]	0x00_2D00_0000	16MB	0x00_2DFF_0000	S/NS
Non-trusted SRAM	0x00_2E00_0000	64KB	0x00_2E00_FFFF	S/NS
GICv3 IRI GICD[w]	0x00_2F00_0000	64KB	0x00_2F00_FFFF	S/NS
GICv3 IRI GITS[w]	0x00_2F02_0000	128KB	0x00_2F03_FFFF	S/NS
GICv3 IRI GICR[w]	0x00_2F10_0000	1MB	0x00_2F1F_FFFF	S/NS
PCIe config region[v]	0x00_4000_0000	256MB	0x00_4FFF_FFFF	S/NS
PCIe memory region 1[v]	0x00_5000_0000	256MB	0x00_5FFF_FFFF	S/NS
Trusted Random Number Generator	0x00_7FE6_0000	4KB	0x00_7FE6_0FFF	S
Trusted Non-volatile counters	0x00_7FE7_0000	4KB	0x00_7FE7_0FFF	S
Trusted Root-Key Storage	0x00_7FE8_0000	4KB	0x00_7FE8_0FFF	S
DDR3 PHY, unimplemented	0x00_7FEF_0000	64KB	0x00_7FEF_FFFF	S/NS
HD LCD Controller, PL370	0x00_7FF6_0000	4KB	0x00_7FF6_0FFF	S/NS
DRAM, 0GB-2GB	0x00_8000_0000	2GB	0x00_FFFF_FFFF	P
DRAM, 2GB-32GB	0x08_8000_0000	30GB	0x0F_FFFF_FFFF	P
PCIe memory region 2[v]	0x40_0000_0000	256GB	0x7F_FFFF_FFFF	S/NS
DRAM, 32GB-512GB	0x88_0000_0000	480GB	0xFF_FFFF_FFFF	P
DRAM, 512GB-8TB	0x00_0880_0000_0000	7TB	0x00_0FFF_FFFF_FFFF	P
DRAM, 8TB-128TB	0x00_8800_0000_0000	120TB	0x00_FFFF_FFFF_FFFF	P
DRAM, 128TB-2PB	0x08_8000_0000_0000	1920TB	0x0F_FFFF_FFFF_FFFF	P
DRAM, 2PB-4PB	0x88_0000_0000_0000	2PB	0x8F_FFFF_FFFF_FFFF	P

Related references

7.3.3 BaseR Platform memory map on page 7-2969

7.3.3 BaseR Platform memory map

The BaseR Platform copies its memory map from the Base Platform, but swaps the upper 2GB of address space with the lower 2GB.

[v] Base Platform RevC only
[w] You can configure the address of this region using parameters to the model. See *3.9.29 GICv3IRI* on page 3-746.
[x] Base Platform RevC AEMv8 FVPs only

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Therefore:

- Any peripherals in the memory range [0x0-0x7FFFFFFF] in the Base Platform are available at the same offset in the memory range [0x80000000-0xFFFFFFFF] in the BaseR Platform.
- Memory in the range [0x80000000-0xFFFFFFFF] in the Base Platform is available at the same offset in the range [0x0-0x7FFFFFFF] in the BaseR Platform. For example, DRAM in the Base Platform memory map starts at address 0x80000000. In the BaseR Platform, this is mapped to 0x0.

The reason for this difference is that in the Armv8-R architecture, the upper 2GB of memory does not have execution permissions by default. So if DRAM in the BaseR Platform memory map started at address 0x80000000, code could not run from DRAM after reset.

Related references

7.3.2 Base Platform memory map on page 7-2967

7.3.4 Base - DRAM

The multiple DRAM regions do not alias each other and form a contiguous 4PB area. The total amount of DRAM on the Base Platform system model is configurable. This ability affects where usable DRAM appears.

If the Base Platform system model has bp.dram_size=4, the default, then 2GB of DRAM is accessible at 0x00_8000_0000 to 0x00_FFFF_FFFF, and the remaining 2GB is accessible at 0x08_8000_0000 to 0x08_FFFF_FFFF.

If, instead, the Base Platform system model has bp.dram_size=8, then 2GB of DRAM is accessible at 0x00_8000_0000 to 0x00_FFFF_FFFF and the remaining 6GB is accessible at 0x08_8000_0000 to 0x09_FFFF_FFFF.

The default contents of RAM not otherwise written by the simulation is a repeating sequence of the following 64-bit value: 0xCFDFDFDFDFDFDFCF.

——— **Note** ———

Memory is allocated on demand, and performance degrades if very large amounts of memory are used.

———————————

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

7.4 Base - interrupt assignments

The platform assigns the *Shared Peripheral Interrupts* (SPIs) and *Private Peripheral Interrupts* (PPIs) on the GIC.

Note

- SPI and PPI numbers are mapped onto GIC interrupt IDs as the *Arm® Generic Interrupt Controller Specification* describes.
- IRQ IDs 103-111 and 200-207 apply to the Base Platform RevC only.

Table 7-6 SPI GIC assignments

IRQ ID	SPI offset	Device
32	0	Watchdog, SP805.
34	2	Dual-Timer 0, SP804.
35	3	Dual-Timer 1, SP804.
36	4	Real-time Clock, PL031.
37	5	UART0, PL011.
38	6	UART1, PL011.
39	7	UART2, PL011.
40	8	UART3, PL011.
41	9	MCI, PL180, MCIINTR0.
42	10	MCI, PL180, MCIINTR1.
43	11	AACI, PL041.
44	12	KMI - Keyboard, PL050.
45	13	KMI - Mouse, PL050.
46	14	Color LCD Controller, PL111.
47	15	Ethernet, SMSC 91C111.
56	24	Trusted Watchdog, SP085.
57	25	AP_REFCLK, Generic Timer, CNTPSIRQ.
58	26	AP_REFCLK, Generic Timer, CNTPSIRQ1.
59	27	EL2 Generic Watchdog WS0.
60	28	EL2 Generic Watchdog WS1.
74	42	Virtio block device.
75	43	Virtio P9 device.
76	44	Virtio net device.
80	48	TZC-400.
92	60	cluster0.cpu0 PMUIRQ.
93	61	cluster0.cpu1 PMUIRQ.
94	62	cluster0.cpu2 PMUIRQ.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 7-6 SPI GIC assignments (continued)

IRQ ID	SPI offset	Device
95	63	cluster0.cpu3 PMUIRQ.
96	64	cluster1.cpu0 PMUIRQ.
97	65	cluster1.cpu1 PMUIRQ.
98	66	cluster1.cpu2 PMUIRQ.
99	67	cluster1.cpu3 PMUIRQ.
103[y]	71	SMMUv3 non-secure combined interrupt.
104[y]	72	SMMUv3 secure combined interrupt. Unused because there is no secure side.
105[y]	73	SMMUv3 secure event queue. Unused because there is no secure side.
106[y]	74	SMMUv3 non-secure event queue.
107[y]	75	SMMUv3 PRI queue. Unused because no PCIe device supports PRI.
108[y]	76	SMMUv3 secure command queue sync. Unused because there is no secure side.
109[y]	77	SMMUv3 non-secure command queue sync.
110[y]	78	SMMUv3 secure GERROR. Unused because there is no secure side.
111[y]	79	SMMUv3 non-secure GERROR.
117	85	HD LCD Controller, PL370.
139	107	Trusted Random Number Generator.
192[z]	160	Mali-G76 GPU.
193[z]	161	Mali-G76 GPU job.
194[z]	162	Mali-G76 GPU MMU.
200[y]	168	PCIe INTA.
201[y]	169	PCIe INTB.
202[y]	170	PCIe INTC.
203[y]	171	PCIe INTD.
207[y]	175	PCIe SERR.

Table 7-7 PPI GIC assignments

IRQ ID	PPI offset	Device
19	3	Secure hypervisor virtual timer interrupt
20	4	Secure hypervisor physical timer interrupt
22	6	DCC, comms channel, interrupt
23	7	PMU, performance counter, overflow
24	8	CTI, Cross Trigger Interface, interrupt
25	9	Virtual CPU interface maintenance interrupt

[y] Base Platform RevC only
[z] Base Platform RevC AEMv8 FVPs only

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 7-7 PPI GIC assignments (continued)

IRQ ID	PPI offset	Device
26	10	Hypervisor timer interrupt
27	11	Virtual timer interrupt
28	12	Hypervisor virtual timer interrupt
29	13	Secure physical timer interrupt
30	14	Non-secure physical timer interrupt

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

7.5 Base - clocks

This section describes the clock frequencies of the Base Platform peripherals.

Table 7-8 Peripheral clock frequencies in the Base Platform

Device	Clock
Clusters	100MHz
REFCLK CNTControl, Generic Timer	100MHz
AP_REFCLK CNTCTL, Generic Timer	100MHz
Dual-Timer 0-1, SP804	35MHz
VE system registers	24MHz
UART 0-3, PL011	24MHz
KMI 0-1, PL050	24MHz
MCI, PL180	24MHz
AACI, PL041	24MHz
Ethernet, SMSC 91C111	24MHz
Watchdog, SP805	24MHz
Color LCD Controller, PL111	23.75MHz
HD LCD Controller, PL370	10MHz
Trusted Watchdog, SP805	32.768kHz
Real-time Clock, PL031	1Hz

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

7.6 Base - parameters

This section describes the parameters.

Table 7-9 Base Platform parameters

Parameter	Type	Allowed values	Default value	Description
`bp.dram_size`	`int`	2, 4, or 8-4000000	4	Size of main memory in gigabytes: 2, 4, or any value between 8 and 4000000.
`bp.proc_id`*n* [aa]	`uint32_t`	-	- [ab]	Processor ID for VE_SysRegs SYS_PROCID*n*.
`bp.secure_memory`	`bool`	`true`, `false`	`true`	The security state of the processor limits access to peripherals and RAM. [ac]
`bp.variant` [aa]	`uint32_t`	`0x0-0xF`	- [ab]	Board variant for VE_SysRegs SYS_ID.
`cache_state_modelled`	`bool`	`true`, `false`	`true`	Enable d-cache and i-cache state for all components.

Table 7-10 Base Platform debug parameters

Parameter	Type	Allowed values	Default value	Description
`dbgen`	`bool`	`true`, `false`	`true`	Debug authentication signal, **dbgen**.
`niden`	`bool`	`true`, `false`	`true`	Debug authentication signal, **niden**.
`spiden`	`bool`	`true`, `false`	`true`	Debug authentication signal, **spiden**.
`spniden`	`bool`	`true`, `false`	`true`	Debug authentication signal, **spniden**.

Related references

VE_SysRegs - parameters on page 7-2982

aa Some platforms do not expose this parameter.
ab Platform specific.
ac When loading an image on an EVS, you might see the following warning:

```
Warning: Base.cluster0.cpu0: Uncaught exception, thread terminated
                 In file: gen/scx_scheduler_mapping.cpp:523
                 In process: Base.thread_p_5 @ 0 s
```

This warning means that the image is attempting to run from DRAM, but this is access-controlled by the TZC_400 component. To disable security checking by the TZC_400, specify `-C Base.bp.secure_memory=false` when running the EVS.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

7.7 Base - components

This section describes the components.

This section contains the following subsections:

- *7.7.1 Base - components - about* on page 7-2976.
- *7.7.2 Base - Base_PowerController component* on page 7-2976.
- *7.7.3 Base - DebugAccessPort component* on page 7-2980.
- *7.7.4 Base - simulator visualization component* on page 7-2981.
- *7.7.5 Base - VE_SysRegs component* on page 7-2982.

7.7.1 Base - components - about

These component models implement some of the functionality of the Versatile Express (VE) hardware.

A complete model implementation of a Base Platform system model includes both Base Platform-specific components and generic components such as buses and timers. To see a list of all the component instances in the model, run it with the `--list-instances` option.

7.7.2 Base - Base_PowerController component

This section describes the Base_PowerController component.

Base_PowerController - control interface

The Base_PowerController provides a basic register interface for software to control the power-up and power-down of cores in the cluster.

Identify cores in the system to the Base_PowerController by writing 24 bits in MPIDR format, providing the following levels of affinity:

Bits [23:16]
Affinity level 2.

Bits [15:8]
Affinity level 1.

Bits [7:0]
Affinity level 0.

Examples of affinity usage are `not_applicable`/`cluster`/`processor` and `cluster`/`processor`/`thread`.

Base_PowerController - parameters

This section describes the parameters.

Table 7-11 Base_PowerController parameters

Parameter	Allowed values	Default value	Description
`startup`	-	'0.0.0.*'	A comma-separated list of cores to power up at startup or system reset. Specify core affinities with a dotted-quad ('0.0.0.0' refers to cluster0.cpu0 and '0.0.1.1' refers to cluster1.cpu1). Use wildcards to indicate all cores at an affinity level ('0.0.0.*' indicates all cores in cluster 0 and is equivalent to '0.0.0.0,0.0.0.1,0.0.0.2,…,0.0.0.255').

Base_PowerController - registers

This section describes the registers.

Register summary

This section describes the power control registers in order of offset from the base memory address.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 7-12 Base_PowerController register summary

Offset	Name	Type	Reset	Width	Description
0x00	PPOFFR	RW	0x---	32	Power Control Processor Off Register
0x04	PPONR	RW	0x---	32	Power Control Processor On Register
0x08	PCOFFR	RW	0x---	32	Power Control Cluster Off Register
0x0C	PWKUPR	RW	0x---	32	Power Control Wakeup Register
0x10	PSYSR	RW	0x---	32	Power Control SYS Status Register

PPOFFR

The *Power Control Processor Off Register* (PPOFFR) characteristics are: purpose, usage constraints, configurations, and attributes.

Purpose

Processor SUSPEND command when PWKUPR and the GIC are programmed appropriately to provide wakeup events from IRQ and FIQ events to that processor.

Usage constraints

Processor must make power-off requests only for itself.

Configurations

Available in all configurations.

Attributes

See the register summary table.

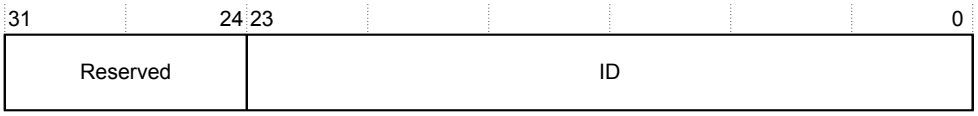

Figure 7-1 Power Control Processor Off Register bit assignments

Table 7-13 Power Control Processor Off Register bit assignments

Bits	Name	Function
[31:24]	-	Reserved.
[23:0]	ID	MPIDR format affinity value of the processor to be switched off. Programming error if MPIDR != self.

PPONR

The *Power Control Processor On Register* (PPONR) characteristics are: purpose, usage constraints, configurations, and attributes.

Purpose

Brings up a processor from low-power mode.

Usage constraints

Processor must make power-on requests only for other powered-off processors in the system.

Configurations

Available in all configurations.

Attributes

See the register summary table.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

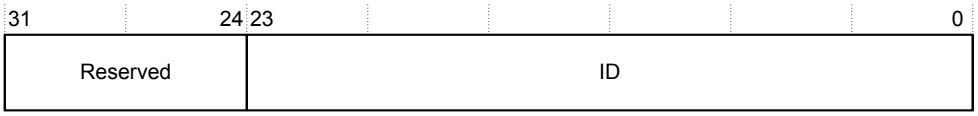

Figure 7-2 Power Control Processor On Register bit assignments

Table 7-14 Power Control Processor On Register bit assignments

Bits	Name	Function
[31:24]	-	Reserved.
[23:0]	ID	MPIDR format affinity value of the processor to be switched on. Programming error if MPIDR == self.

PCOFFR

The *Power Control Cluster Off Register* (PCOFFR) characteristics are: purpose, usage constraints, configurations, and attributes.

Purpose

Turns the cluster off.

Usage constraints

Cluster must make power-off requests only for itself.

Configurations

Available in all configurations.

Attributes

See the register summary table.

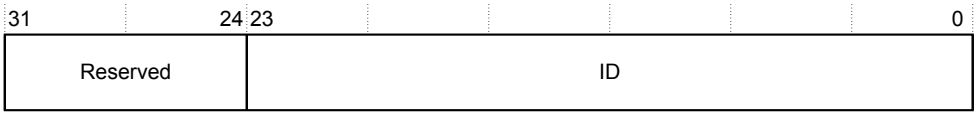

Figure 7-3 Power Control Cluster Off Register bit assignments

Table 7-15 Power Control Cluster Off Register bit assignments

Bits	Name	Function
[31:24]	-	Reserved.
[23:0]	ID	MPIDR format affinity value of powered-on processor in the cluster to be switched off. Programming error if MPIDR != self.

PWKUPR

The *Power Control Wakeup Register* (PWKUPR) characteristics are: purpose, usage constraints, configurations, and attributes.

Purpose

Configures whether wakeup requests from the GIC are enabled for this cluster.

Usage constraints

There are no usage constraints.

Configurations

Available in all configurations.

Attributes

See the register summary table.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

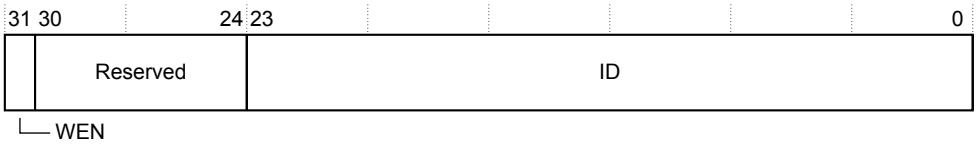

Figure 7-4 Power Control Wakeup Register bit assignments

Table 7-16 Power Control Wakeup Register bit assignments

Bits	Name	Function
[31]	WEN	If set, enables wakeup interrupts (return from SUSPEND) for this cluster.
[30:24]	-	Reserved.
[23:0]	ID	MPIDR format affinity value of processor whose Wakeup Enable bit is to be configured.

PSYSR

The *Power Control SYS Status Register* (PSYSR) characteristics are: purpose, usage constraints, configurations, and attributes.

Purpose

Provides information on the powered status of a given core. Software writes bits [23:0] for the required core and reads the value along with the associated status in bits [31:24].

Usage constraints

There are no usage constraints.

Configurations

Available in all configurations.

Attributes

See the register summary table.

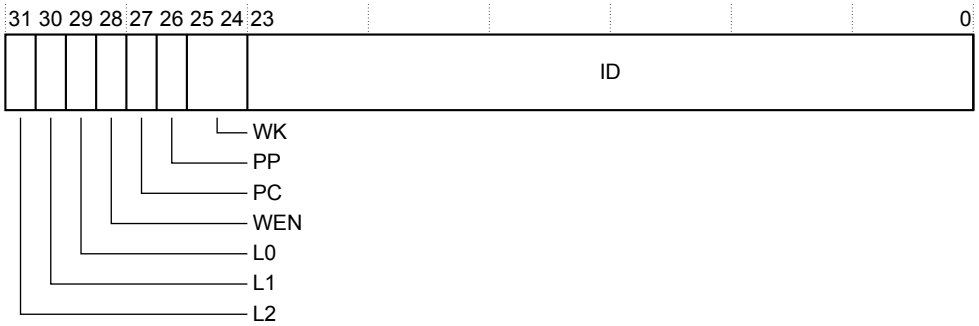

Figure 7-5 Power Control SYS Status Register bit assignments

Table 7-17 Power Control SYS Status Register bit assignments

Bits	Name	Function
[31]	L2	Read-only. A value of 1 indicates that affinity level 2 is active/on. If affinity level 2 is not implemented this bit is RAZ.
[30]	L1	Read-only. A value of 1 indicates that affinity level 1 is active/on. If affinity level 1 is not implemented this bit is RAZ.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 7-17 Power Control SYS Status Register bit assignments (continued)

Bits	Name	Function
[29]	L0	Read-only. A value of 1 indicates that affinity level 0 is active/on.
[28]	WEN	Read-only. A value of 1 indicates wakeup interrupts, return from SUSPEND, enabled for this processor. This is an alias of PWKUPR.WEN for this core.
[27]	PC	Read-only. A value of 1 indicates pending cluster off, the cluster enters low-power mode the next time it raises signal STANDBYWFIL2.
[26]	PP	Read-only. A value of 1 indicates pending processor off, the processor enters low-power mode the next time it raises signal STANDBYWFI.
[25:24]	WK	Read-only. Indicates the reason for LEVEL0 power on: **0b00** Cold power-on. **0b01** System reset pin. **0b10** Wake by PPONR. **0b11** Wake by GIC WakeRequest signal.
[23:0]	ID	MPIDR format affinity value.

7.7.3 Base - DebugAccessPort component

This section describes the DebugAccessPort component, a model of the *Debug Access Port* (DAP) for external debug connections.

DebugAccessPort - ports

This section describes the ports.

Table 7-18 DebugAccessPort ports

Name	Protocol	Type	Description
`ap_pvbus_m[2]`	PVBus	Master	Debug-access port to bus master channels 0 and 1.
`clock`	ClockSignal	Slave	Clock input.
`paddrdbg31`	Signal	Master	Output signal that indicates which master the access came from, AP0 or AP1. Configurable.

DebugAccessPort - parameters

This section describes the parameters.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 7-19 Base Platform DebugAccessPort parameters

Name	Type	Allowed values	Default value	Description
ap0_rom_base_address	uint64_t	0x0-0xffffffffffffffff	0x0	ROM base address for AP0.
ap1_rom_base_address	uint64_t	0x0-0xffffffffffffffff	0x0	ROM base address for AP1.
ap0_has_debug_rom	bool	true, false	false	AP0 has a Debug ROM.
ap1_has_debug_rom	bool	true, false	false	AP1 has a Debug ROM.
ap0_set_paddrdbg31	bool	true, false	false	Set paddrdbg31 signal during accesses on AP0.
ap1_set_paddrdbg31	bool	true, false	false	Set paddrdbg31 signal during accesses on AP1.

7.7.4 Base - simulator visualization component

This section describes the simulator visualization component.

Simulator visualization - parameters

This section describes the parameters.

Table 7-20 Simulator visualization parameters

Parameter	Allowed values	Default value	Description
cluster0_name	-	"Cluster0"	Cluster0 name.
cluster1_name	-	"Cluster1"	Cluster1 name.
cpu_name	-	""	Window title displays core name.
disable_visualisation	true, false	false	Enables or disables visualization.
rate_limit-enable	true, false	true	Rate limit simulation.
recorder.checkInstructionCount	true, false	true	Checks instruction count in recording file against actual instruction count during playback.
recorder.playbackFileName	-	""	Playback filename. An empty string disables playback.
recorder.recordingFileName	-	""	Recording filename. An empty string disables recording.
recorder.recordingTimeBase	-	0x5F5E100	Timebase in 1/s relative to the master clock. For example, 100 000 000 means 10 nanoseconds resolution simulated time for a 1Hz master clock. Used for recording. Higher values give a higher time resolution. Playback time base is always taken from the playback file.
recorder.verbose	-	0x0	Enables verbose messages. 0x0, no messages. 0x1, normal. 0x2, more detail.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 7-20 Simulator visualization parameters (continued)

Parameter	Allowed values	Default value	Description
`trap_key`	-	`0x4a`	Trap key that works with left **`Ctrl`** to toggle mouse display.
`window_title`	-	"Fast Models - CLCD %cpu%"	`cpu_name` replaces window title, `%cpu%`.

7.7.5 Base - VE_SysRegs component

This section describes the VE system registers component.

VE_SysRegs - parameters

This section describes the parameters.

Table 7-21 Base Platform VE_SysRegs parameters

Name	Type	Allowed values	Default value	Description
`exit_on_shutdown`	`bool`	`true`, `false`	`false`	`true`: if software uses the `SYS_CFGCTRL` function `SYS_CFG_SHUTDOWN`, then the simulator shuts down and exits.[ad]
`mmbSiteDefault`	`uint8_t`	`0x0-0x2`	`0x1`	Default MMB source. `0x0`, MB. `0x1`, DB1. `0x2`, DB2.
`tilePresent`	`bool`	`true`, `false`	`true`	Tile fitted.
`user_switches_value`	`uint32_t`	-	`0x0`	User switches.

Related references

7.6 Base - parameters on page 7-2975

VE_SysRegs - registers

This section describes the configuration registers.

Table 7-22 Base Platform VE_SysRegs registers

Name	Offset	Access	Description
SYS_ID	`0x00`	Read/write	System identity.
SYS_SW	`0x04`	Read/write	Bits[7:0] map to switch S6.
SYS_LED	`0x08`	Read/write	Bits[7:0] map to user LEDs.
SYS_100HZ	`0x24`	Read only	100Hz counter.
SYS_FLAGS	`0x30`	Read/write	General-purpose flags.
SYS_FLAGSCLR	`0x34`	Write only	Clear bits in general-purpose flags.
SYS_NVFLAGS	`0x38`	Read/write	General-purpose non-volatile flags.
SYS_NVFLAGSCLR	`0x3C`	Write only	Clear bits in general-purpose non-volatile flags.
SYS_MCI	`0x48`	Read only	MCI.
SYS_FLASH	`0x4C`	Read/write	Flash control.

[ad] For more information on `SYS_CFGCTRL`, see the *Motherboard Express µATX V2M-P1 Technical Reference Manual*.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 7-22 Base Platform VE_SysRegs registers (continued)

Name	Offset	Access	Description
SYS_CFGSW	0x58	Read/write	Boot-select switch.
SYS_24MHZ	0x5C	Read only	24MHz counter.
SYS_MISC	0x60	Read/write	Miscellaneous control flags.
SYS_DMA	0x64	Read/write	DMA peripheral map.
SYS_PROCID0	0x84	Read/write	Processor ID.
SYS_PROCID1	0x88	Read/write	Processor ID.
SYS_PROCID2	0x8C	Read/write	Processor ID.
SYS_PROCID3	0x90	Read/write	Processor ID.
SYS_CFGDATA	0xA0	Read/write	Data to read/write from & to motherboard controller.
SYS_CFGCTRL	0xA4	Read/write	Control data transfer to motherboard controller.
SYS_CFGSTAT	0xA8	Read/write	Status of data transfer to motherboard controller.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

7.8 Base - differences between the AEMv8-A FVP and core FVPs

This section describes implementation features of the core models that the AEMv8-A model does not implement, or implements with significant differences.

- The default value of `cache_state_modelled` is 0.
- Components `cluster0` and `cluster1` are implementation cores, not AEMs. All parameters in these components are the parameters of the core implementation, not the parameters of the AEM. The values of `bp.proc_id0` and `bp.proc_id1` have fixed values consistent with the cores and are not configurable.
- The core defines the memory map of register banks within the GIC region, and the map is therefore not configurable. The parameter `bp.variant` communicates the nature of the memory map to target software. The parameter has a fixed value consistent with the memory map of the core, and is not configurable. Because the core implementations contain a specific version of the GIC, the parameter `gicv3.gicv2-only` is not available. The following registers in the GIC distributor are given fixed values to match the implementation, and are not configurable: `gic_distributor.reg-base`, `gic_distributor.reg-base-per-distributor`, `gic_distributor.GICD-alias`, `gic_distributor.ITS0-base`.
- The GICv2 CPU IDs are contiguous in implementation platform models, because the number of cores in each cluster is fixed. They can be non-contiguous in the AEMv8-A Base Platform FVP because there is space for four cores in the cluster. You can configure fewer than four cores.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

7.9 Base - VE compatibility

Arm expects software that ran on the previous VE model to be compatible with this system model, but you might need to apply some configuration options.

This section contains the following subsections:

- *7.9.1 Base - VE compatibility - GICv2* on page 7-2985.
- *7.9.2 Base - VE compatibility - GICv3* on page 7-2985.
- *7.9.3 Base - VE compatibility - system global counter* on page 7-2986.
- *7.9.4 Base - VE compatibility - disable security* on page 7-2986.

7.9.1 Base - VE compatibility - GICv2

This system model uses GICv3 by default. You can configure it to support GICv2 or GICv2m.

To configure the model as GICv2m, set the following:

```
-C gicv3.gicv2-only=1                          \
-C cluster0.gic.GICD-offset=0x10000            \
-C cluster0.gic.GICC-offset=0x2F000            \
-C cluster0.gic.GICH-offset=0x4F000            \
-C cluster0.gic.GICH-other-CPU-offset=0x50000  \
-C cluster0.gic.GICV-offset=0x6F000            \
-C cluster0.gic.PERIPH-size=0x80000            \
-C cluster1.gic.GICD-offset=0x10000            \
-C cluster1.gic.GICC-offset=0x2F000            \
-C cluster1.gic.GICH-offset=0x4F000            \
-C cluster1.gic.GICH-other-CPU-offset=0x50000  \
-C cluster1.gic.GICV-offset=0x6F000            \
-C cluster1.gic.PERIPH-size=0x80000            \
-C gic_distributor.GICD-alias=0x2c010000
```

To configure the model as GICv2, set the following:

```
-C gicv3.gicv2-only=1                         \
-C cluster0.gic.GICD-offset=0x1000            \
-C cluster0.gic.GICC-offset=0x2000            \
-C cluster0.gic.GICH-offset=0x4000            \
-C cluster0.gic.GICH-other-CPU-offset=0x5000  \
-C cluster0.gic.GICV-offset=0x6000            \
-C cluster0.gic.PERIPH-size=0x8000            \
-C cluster1.gic.GICD-offset=0x1000            \
-C cluster1.gic.GICC-offset=0x2000            \
-C cluster1.gic.GICH-offset=0x4000            \
-C cluster1.gic.GICH-other-CPU-offset=0x5000  \
-C cluster1.gic.GICV-offset=0x6000            \
-C cluster1.gic.PERIPH-size=0x8000            \
-C gic_distributor.GICD-alias=0x2c010000
```

To configure MSI frames for GICv2m, parameters are available to set the base address and configuration of each of 16 possible frames. Eight frames are Secure and eight frames are Non-secure:

```
-C gic_distributor.MSI_S-frame0-base=ADDRESS  \
-C gic_distributor.MSI_S-frame0-min-SPI=NUM   \
-C gic_distributor.MSI_S-frame0-max-SPI=NUM
```

In this example, you can replace `MSI_S` with MSI_NS, for NS frames, and you can replace `frame0` with `frame1` to `frame7` for each of the possible 16 frames. If the base address is not specified for a given frame, or the SPI numbers are out of range, the corresponding frame is not instantiated.

7.9.2 Base - VE compatibility - GICv3

If a Base Platform includes an implementation of the GICv3 system registers, it is enabled by default.

The GIC distributor and CPU (core) interface have parameters that allow configuration of the model to match different implementation options. Use `--list-params` to get a full list. Configuration options for the GIC model must be available under:

- `cluster[0-`*n*`].gic.*`
- `cluster[0-`*n*`].gicv3.*`
- `gic_distributor.*`

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

7.9.3 Base - VE compatibility - system global counter

The Generic Timer registers of the cores do not operate by default.

The model provides a memory-mapped interface to the system global counter, and enables the free-running timer from reset. However, the architectural requirement is that such a counter is not enabled at reset. As a result, the Generic Timer registers of the cores do not operate unless either:

- Software enables the counter peripheral by writing the FCREQ[0] and EN bits in CNTCR at `0x2a430000`. Arm recommends this approach.
- The `-C bp.refcounter.non_arch_start_at_default=1` parameter is set. This approach provides compatibility with older software.

7.9.4 Base - VE compatibility - disable security

Base Platform FVPs have an enhanced security map for peripherals. By default, it restricts access to some peripherals.

Software must program the TZC-400 to make any accesses to DRAM, because the reset configuration blocks all accesses.

For backward compatibility with software that cannot program the TZC-400, this parameter setting permits all accesses regardless of security state:

```
-C bp.secure_memory=false
```

Related references

7.3.2 Base Platform memory map on page 7-2967

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

7.10 Base - unsupported VE features

This system model does not support software that relies on some features of the VE model.

This section contains the following subsections:

- *7.10.1 Base - unsupported VE features - memory aliasing at 0x08_00000000* on page 7-2987.
- *7.10.2 Base - unsupported VE features - boot ROM alias at 0x00_0800_0000* on page 7-2987.
- *7.10.3 Base - unsupported VE features - change of older parameters* on page 7-2987.

7.10.1 Base - unsupported VE features - memory aliasing at 0x08_00000000

The VE model permits an alias of the 2GB region of DRAM between addresses `0x80000000` and `0xFFFFFFFF` with addresses `0x08_00000000` to `0x08_7FFFFFFF`. The Base Platform does not have this alias and the region `0x08_00000000` to `0x08_7FFFFFFF` is Reserved.

7.10.2 Base - unsupported VE features - boot ROM alias at 0x00_0800_0000

In the VE model, the region at `0x00_0800_0000` was an alias of the trusted boot ROM at `0x00_0000_0000`. It is now an independent region of NOR flash.

7.10.3 Base - unsupported VE features - change of older parameters

Most parameter names have been simplified between the VE model and the Base Platform system model.

Components that were previously in *motherboard* or *daughterboard* groups are now in a *bp* group. The model does not recognize the previous parameter names.

In a change to the previous default, the Base Platform models the core cache state by default. You can disable this using a single parameter for all cores in the simulation, using the `cache_state_modelled` parameter.

```
-C cache_state_modelled=0
```

Note

Cortex Base Platforms do not model the cache state by default.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Chapter 8
Microcontroller Prototyping System 2

This chapter describes the MPS2 system model.

It contains the following sections:

- *8.1 MPS2 - about* on page 8-2989.
- *8.2 MPS2 platform types* on page 8-2990.
- *8.3 MPS2 - memory maps* on page 8-2991.
- *8.4 MPS2 - interrupt assignments* on page 8-2997.
- *8.5 MPS2 - differences between models and hardware* on page 8-2998.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

8.1 MPS2 - about

The *Microcontroller Prototyping System 2* (MPS2) *Fixed Virtual Platform* (FVP) models implement a subset of the functionality of the Arm Versatile Express V2M-MPS2 and V2M-MPS2+ motherboard hardware.

For a list of the MPS2 Platform FVPs and the instances in them, see *MPS2 Platform FVPs* in the FVP Reference Guide.

LISA+ source and project files for some MPS2 FVP examples are available in the `$PVLIB_HOME/examples/LISA/FVP_MPS2/` directory.

MPS2 platforms include MPS2-specific components and generic components, such as buses and timers. They are sufficiently accurate to boot the Keil® RTX RTOS and run the Blinky application.

To list the model parameters and their types, allowed values, default values, and descriptions, run the model with the `--list-params` argument.

Related information

AN400 - Arm Cortex-M7 SMM on V2M-MPS2

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

8.2 MPS2 platform types

Configure the MPS2 FVP platform type using the `fvp_mps2.platform_type` parameter.

It has the following possible values:

0 This value is the default. The FVP acts as a V2M-MPS2 system, with the additions for v8-M, as specified in the Armv8-M MPS2 System Specification (ECM 0468897), v0.8. This specification is confidential and is available only to licensed Arm customers. For details, contact your Arm support representative.

1 The FVP acts as an IoT Kit on an MPS2+ board. For details, see the following documents:

- *Cortex®-M23 processor Arm®v8-M IoT Kit User Guide (ECM 0635473)*.
- *Cortex®-M33 processor Arm®v8-M IoT Kit User Guide (ECM 0601256)*.

2 The FVP acts as an Arm CoreLink SSE-200 Subsystem on an MPS2+ board. For details, see *AN521 - Example SSE-200 Subsystem for MPS2+ Application Note*.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

8.3 MPS2 - memory maps

This section describes the MPS2 memory maps.

This section contains the following subsections:

- *8.3.1 MPS2 - memory map for models without the Arm®v8-M additions* on page 8-2991.
- *8.3.2 MPS2 - memory map for models with the Arm®v8-M additions* on page 8-2992.

8.3.1 MPS2 - memory map for models without the Arm®v8-M additions

This section describes the MPS2 memory map for older cores, without the Armv8-M additions.

For standard Arm peripherals, see the TRM for that device.

Note

- A bus error is generated for accesses to memory areas not shown in this table.
- Any memory device that does not occupy the total region is aliased within that region.

Table 8-1 Overview of MPS2 memory map

Description	Modeled	Address range
Ethernet[ae]	Partial	0xA0000000-0xA000FFFF
PSRAM (16MB)	Yes	0x60000000-0x60FFFFFF
VGA Image (512x128) (AHB)	Yes	0x41100000-0x4110FFFF
VGA Console (AHB)	Yes	0x41000000-0x4100FFFF
Block RAM (boot time)[af]	Yes	0x40200000-0x402FFFFF
Reserved	N/A	0x40030000-0x401FFFFF
SCC register	Yes	0x4002F000-0x4002FFFF
Reserved	N/A	0x40029000-0x4002EFFF
FPGA System Control & I/O, APB	Yes	0x40028000-0x40028FFF
Reserved	N/A	0x40025000-0x40027FFF
Audio I2S, APB	Partial	0x40024000-0x40024FFF
SBCon (Audio Configuration), APB	Yes	0x40023000-0x40023FFF
SBCon (Touch for LCD module), APB	Partial	0x40022000-0x40022FFF
PL022 (SPI for LCD module), APB	Partial	0x40021000-0x40021FFF
PL022 (SPI), APB	Yes	0x40020000-0x40020FFF
CMSDK system controller	Yes	0x4001F000-0x4001FFFF
Reserved for extra GPIO & other AHB peripherals	N/A	0x40012000-0x4001EFFF
CMSDK AHB GPIO #1	Yes	0x40011000-0x40011FFF
CMSDK AHB GPIO #0	Yes	0x40010000-0x40010FFF
CMSDK APB subsystem	Yes	0x40000000-0x4000FFFF
Reserved	N/A	0x20800000-0x20FFFFFF

ae Through ahb_to_extmem16. Offset 0x0-0x0FE for CSRs, 0x100-0x1FE for FIFO.
af Reserved 64KB, 16K implemented. This memory is wrapped through the region.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 8-1 Overview of MPS2 memory map (continued)

Description	Modeled	Address range
ZBTSRAM 2 & 3 (2x32-bit)[ag]	Yes	`0x20000000-0x207FFFFF`
Reserved	N/A	`0x01010000-0x1FFFFFFF`
Reserved	N/A	`0x00800000-0x00FFFFFF`
ZBTSRAM 1 (64-bit)[ah]	Yes	`0x00400000-0x007FFFFF`
ZBTSRAM 1 (64-bit)	Yes	`0x00004000-0x003FFFFF`
Mappable memory[ai]	Yes	`0x00000000-0x00003FFF`

8.3.2 MPS2 - memory map for models with the Arm®v8-M additions

This section describes the MPS2 memory map for newer cores, with the Armv8-M additions.

For standard Arm peripherals, see the TRM for that device.

Note

- A bus error is generated for accesses to memory areas not shown in this table.
- Any memory device that does not occupy the total region is aliased within that region.

Table 8-2 Overview of MPS2 memory map

Description	IDAU	Modeled	Address range
ZBTSRAM 1 (4MB) in *Non-secure* (NS) world. Reserved 8MB, only 4MB implemented. VTOR initialization value to be configurable in LAC). Second half (4MB) aliased to first half (4MB).	NS	Yes	`0x00000000-0x007FFFFF`
Not used. Default expansion port (MPS2 AHB subsystem): bus error or RAZ/WI.	NS	N/A	`0x00800000-0x0FFFFFFF`
ZBTSRAM 1 (4MB) in *Secure* (S) world. Reserved 8MB, only 4MB implemented. Second half (4MB) aliased to first half (4MB).	S	Yes	`0x10000000-0x107FFFFF`
Not used. Default expansion port: bus error.	S	N/A	`0x10800000-0x1FFFFFFF`
ZBTSRAM 2&3 (4MB) in NS world. Reserved 8MB, only 4MB implemented. For IoT subsystems, different cores have different memory sizes. Second half (4MB) aliased to first half (4MB).	NS	Yes	`0x20000000-0x207FFFFF`
Not used. Default expansion port (MPS2 AHB subsystem): bus error or RAZ/WI.	NS	N/A	`0x20800000-0x20FFFFFF`
PSRAM (16MB)	NS	Yes	`0x21000000-0x21FFFFFF`
Not used. Default expansion port (MPS2 AHB subsystem): bus error or RAZ/WI.	NS	N/A	`0x22000000-0x23FFFFFF`
MTB SRAM. Reserved 64KB, only 16KB implemented. Aliased to `0x0` for booting in RTL simulation.	NS	Yes	`0x24000000-0x2400FFFF`
Not used. Default expansion port (MPS2 AHB subsystem): bus error or RAZ/WI.	NS	N/A	`0x24010000-0x2FFFFFFF`

[ag] Reserved 8MB, 4MB available. The two SRAM blocks are interleaved.
[ah] Wrapped. Only 4MB ZBTSRAM fitted.
[ai] When `zbt_boot_ctrl` = 0, ZBTSRAM 1 is mapped to this region. Otherwise, `Remap_ctrl` = 0 maps Block RAM and `Remap_ctrl` = 1 maps ZBTSRAM 1. The V2M-MPS2 board microcontroller controls the zbt_boot_ctrl signal. The zbt_boot_ctrl signal overrides the boot option to enable use of the ZBT RAM.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 8-2 Overview of MPS2 memory map (continued)

Description	IDAU	Modeled	Address range
ZBTSRAM 2&3 (4MB) in S world. Reserved 8MB, only 4MB implemented. Second half (4MB) aliased to first half (4MB).	S	Yes	0x30000000-0x307FFFFF
Not used. Default expansion port: bus error.	S	N/A	0x30800000-0x30FFFFFF
Not used. No memory gating unit on PSRAM (16MB) path because it is shared with Ethernet control. Default expansion port: bus error.	S	N/A	0x31000000-0x31FFFFFF
Not used. Default expansion port: bus error.	S	N/A	0x30800000-0x3FFFFFFF
CMSDK APB subsystem:	NS	-	0x40000000-0x4000FFFF
Timer 0.	NS	Yes	0x40000000-0x40000FFF
Timer 1.	NS	Yes	0x40001000-0x40001FFF
Dual Timer.	NS	Yes	0x40002000-0x40002FFF
Not used.	NS	N/A	0x40003000-0x40003FFF
UART #0.	NS	Yes	0x40004000-0x40004FFF
UART #1.	NS	Yes	0x40005000-0x40005FFF
UART #2.	NS	Yes	0x40006000-0x40006FFF
Not used.	NS	N/A	0x40007000-0x40007FFF
Watchdog.	NS	Yes	0x40008000-0x40008FFF
Not used.	NS	N/A	0x40009000-0x4000F000
GPIO #0.	NS	Yes	0x40010000-0x40010FFF
GPIO #1.	NS	Yes	0x40011000-0x40011FFF
GPIO #2.	NS	Yes	0x40012000-0x40012FFF
GPIO #3.	NS	Yes	0x40013000-0x40013FFF
Default slave inside AHB peripheral subsystem. Default expansion port (MPS2 AHB subsystem): bus error or RAZ/WI.	NS	Yes	0x40014000-0x40017FFF
DMA Controller #0.	NS	Yes	0x40018000-0x40018FFF
DMA Controller #1.	NS	Yes	0x40019000-0x40019FFF
Default slave inside AHB peripheral subsystem. Default expansion port (MPS2 AHB subsystem): bus error or RAZ/WI.	NS	Yes	0x4001A000-0x40017FFF
CMSDK system controller. PMU control and remap registers unused. Only reset option (lockup reset) and rest info available.	NS	Yes	0x4001F000-0x4001FFFF
FPGA APB Subsystem (unused APB space: RAZ/WI):	NS	-	0x40020000-0x4002FFFF
PL022 (SPI).	NS	Yes	0x40020000-0x40020FFF
PL022 (SPI for LCD).	NS	Partial	0x40021000-0x40021FFF
SBCon I2C (Touch for LCD).	NS	Partial	0x40022000-0x40022FFF
SBCon I2C (Audio configuration).	NS	Yes	0x40023000-0x40023FFF

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 8-2 Overview of MPS2 memory map (continued)

Description	IDAU	Modeled	Address range
Audio I2S.	NS	Partial	0x40024000-0x40024FFF
Not used.	NS	N/A	0x40025000-0x40027FFF
FPGA system control & I/O (LEDs, buttons...).	NS	Yes	0x40028000-0x40028FFF
Not used.	NS	N/A	0x40029000-0x4002EFFF
SCC registers.	NS	Yes	0x4002F000-0x401FFFFF
Ethernet.	NS	Partial	0x40200000-0x402FFFFF
Not used. Default expansion port (MPS2 AHB subsystem): bus error or RAZ/WI.	NS	N/A	0x40300000-0x40FFFFFF
VGA console.	NS	Yes	0x41000000-0x4100FFFF
VGA image.	NS	Yes	0x41100000-0x4113FFFF
Not used. Default expansion port (MPS2 AHB subsystem): bus error or RAZ/WI.	NS	N/A	0x41140000-0x4FFFFFFF
CMSDK APB subsystem:	S	-	0x50000000-0x5000FFFF
Timer 0.	S	Yes	0x50000000-0x50000FFF
Timer 1.	S	Yes	0x50001000-0x50001FFF
Dual Timer.	S	Yes	0x50002000-0x50002FFF
Not used.	S	N/A	0x50003000-0x50003FFF
UART #0.	S	Yes	0x50004000-0x50004FFF
UART #1.	S	Yes	0x50005000-0x50005FFF
UART #2.	S	Yes	0x50006000-0x50006FFF
Not used.	S	N/A	0x50007000-0x50007FFF
Watchdog.	S	Yes	0x50008000-0x50008FFF
Not used.	S	N/A	0x50009000-0x5000F000
GPIO #0	S	Yes	0x50010000-0x50010FFF
GPIO #1	S	Yes	0x50011000-0x50011FFF
GPIO #2	S	Yes	0x50012000-0x50012FFF
GPIO #3	S	Yes	0x50013000-0x50013FFF
Default slave. Default expansion port: bus error.	S	Yes	0x50014000-0x50017FFF
DMA Controller #0.	S	Yes	0x50018000-0x50018FFF
DMA Controller #1.	S	Yes	0x50019000-0x50019FFF
Default slave. Default expansion port: bus error.	S	Yes	0x5001A000-0x5001EFFF
CMSDK system controller. PMU control and remap registers unused. Only reset option (lockup reset) and rest info available.	S	Yes	0x5001F000-0x5001FFFF

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 8-2 Overview of MPS2 memory map (continued)

Description	IDAU	Modeled	Address range
FPGA APB subsystem:	S	-	0x50020000-0x5002FFFF
PL022 (SPI).	S	Yes	0x50020000-0x50020FFF
PL022 (SPI for LCD).	S	Partial	0x50021000-0x50021FFF
SBCon I2C (touch for LCD).	S	Partial	0x50022000-0x50022FFF
SBCon I2C (audio configuration).	S	Yes	0x50023000-0x50023FFF
Audio I2S.	S	Partial	0x50024000-0x50024FFF
Not used.	S	N/A	0x50025000-0x50027FFF
FPGA system control & I/O (LEDs, buttons...).	S	Yes	0x50028000-0x50028FFF
Not used.	S	N/A	0x50029000-0x5002EFFF
SCC registers.	S	Yes	0x5002F000-0x501FFFFF
Ethernet.	S	Partial	0x50200000-0x502FFFFF
Not used. Default expansion port: bus error.	S	N/A	0x50300000-0x50FFFFFF
VGA console.	S	Yes	0x51000000-0x5100FFFF
VGA image.	S	Yes	0x51100000-0x5113FFFF
Not used. Default expansion port: bus error.	S	N/A	0x51140000-0x57FFFFFFS
Secure APB subsystem:	S	-	0x58000000-0x5800FFFF
TRNG / PRNG.	S	Yes	0x58000000-0x58000FFF
Unique ID or Secure storage.	S	Yes	0x58001000-0x58006FFF
Secure Control Registers.	S	Yes	0x58007000-0x58007FFF
Flash memory gating unit configuration (mapped to AHB port for CODE region in the bus matrix, not APB).	S	Yes	0x58008000-0x58009FFF
SRAM memory gating unit configuration (mapped to AHB port for SRAM region in the bus matrix, not APB).	S	Yes	0x5800A000-0x5800DFFF
Reserved.	S	N/A	0x5800E000-0x5800EFFF
Reserved.	S	N/A	0x5800F000-0x5800FFFF
Not used. Default expansion port: bus error.	S	N/A	0x50010000-0x5FFFFFFF
Not used. Default expansion port (MPS2 AHB subsystem): bus error or RAZ/WI.	NS	N/A	0x60000000-0x6FFFFFFF
Not used. Default expansion port: bus error.	S	N/A	0x70000000-0x7FFFFFFF
Not used. Default expansion port (MPS2 AHB subsystem): bus error or RAZ/WI.	NS	N/A	0x800000000-0x8FFFFFFF
Not used. Default expansion port: bus error.	S	N/A	0x90000000-0x9FFFFFFF
Not used. Default expansion port (MPS2 AHB subsystem): bus error or RAZ/WI.	NS	N/A	0xA0000000-0xAFFFFFFF

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 8-2 Overview of MPS2 memory map (continued)

Description	IDAU	Modeled	Address range
Not used. Default expansion port: bus error.	S	N/A	0xB0000000-0xBFFFFFFF
Not used. Default expansion port (MPS2 AHB subsystem): bus error or RAZ/WI.	NS	N/A	0xC0000000-0xCFFFFFFF
Not used. Default expansion port: bus error.	S	N/A	0xD0000000-0xDFFFFFFF
Not used. Default expansion port (MPS2 AHB subsystem): bus error or RAZ/WI.	NS	N/A	0xE0000000-0xEFFFFFFF
System ROM table. Exempted from checking.	Exempt	Yes	0xF0000000-0xF0000FFF
Not used. Default expansion port: bus error.	Exempt	N/A	0xF0001000-0xF00FFFFF
Not used. Default expansion port (MPS2 AHB subsystem): bus error or RAZ/WI.	NS	N/A	0xF0100000-0xF01FFFFF
MTB SFR address space.	NS	Yes	0xF0200000-0xF0200FFF
Reserved. This region is non-executable.	NS	N/A	0xF0210000-0xF0213FFF
Not used. Default expansion port (MPS2 AHB subsystem): bus error or RAZ/WI.	NS	N/A	0xF0214000-0xFFFFFFFF

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

8.4 MPS2 - interrupt assignments

This section describes the interrupt assignments.

Table 8-3 Interrupt assignments

Number	Interrupt
NMI	Watchdog.
0	UART 0 receive interrupt.
1	UART 0 transmit interrupt.
2	UART 1 receive interrupt.
3	UART 1 transmit interrupt.
4	UART 2 receive interrupt.
5	UART 2 transmit interrupt.
6	GPIO 0, 2 combined interrupt.
7	GPIO 1, 3 combined interrupt.
8	Timer 0.
9	Timer 1.
10	Dual Timer.
11	SPI #1 (LCD). The LCD had shared SPI #0 and SPI #1.
12	UART overflow (0, 1, 2).
13	Ethernet.
14	Audio I2S.
15	Touch screen.
16-31	GPIO 0 individual interrupts.
32-47	GPIO 1 individual interrupts. Armv8-M additions.
48	SPI #0. Armv8-M addition.
49	Reserved.
50	TRNG (Secure). Armv8-M addition.
51	Unique ID and Secure storage (Secure). Armv8-M addition.
52	DMA controller #0.
53	DMA controller #1.
54	SecureErrorIRQ. Armv8-M addition.[aj]

[aj] Detection of Non-secure access to Secure address spaces (including other bus masters). Generated by Memory Gating unit, Peripheral Gating units, bus gasket for legacy bus masters.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

8.5 MPS2 - differences between models and hardware

This section describes the features of the hardware that the models do not implement, or implement with significant differences.

MPS2 implements most devices. Some peripherals have minimal implementations:

- The Ethernet module in the model is a LAN91C111. The hardware provides a LAN9220.
- The Audio module is RAZ/WI.
- The STMPE811 touchscreen module only reports touch positions.
- The model of the Ampire LCD module supports a subset of the graphics modes.

You can display images and text on an emulated VGA output, images on the LCD, and text on the UART.

Arm®v8-M

The model does not support MTB, ETM, and TPIU. MTB RAM is absent.

In the Memory Gating Unit, the model provides a configurable block size. For performance reasons, the minimum block size in the model is 4096 bytes. Hardware and later models might allow smaller block sizes. Software must use the BLK_CFG register to determine block size.

Timing

FVPs enable software applications to run in a functionally accurate simulation. However, because of the relative balance of fast simulation speed over timing accuracy, there are situations where the models might behave unexpectedly.

If your code interacts with real world devices such as timers and keyboards, data arrives in the modeled device in real world, or wall clock, time. However, simulation time can run faster than the wall clock. So, a single key press might be interpreted as several repeated key presses, or a single mouse click might be interpreted as a double click.

To avoid this mismatch, the FVPs provide the Rate Limit feature. Enabling Rate Limit forces the model to run at wall clock time. For interactive applications, Arm recommends enabling Rate Limit. Use the Rate Limit button in the CLCD display or the `rate_limit-enable` model instantiation parameter.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Chapter 9
Versatile Express Model

This chapter describes the components of Fast Models that are specific to the Versatile Express (VE) model of the hardware platform.

It contains the following sections:

- *9.1 VE - about* on page 9-3000.
- *9.2 VE memory map for Cortex®-A series* on page 9-3002.
- *9.3 VE memory map for Cortex®-R series* on page 9-3005.
- *9.4 VE - interrupt assignments for Cortex®-A series* on page 9-3006.
- *9.5 VE - interrupt assignments for Cortex®-R series* on page 9-3007.
- *9.6 VE parameters* on page 9-3009.
- *9.7 VE - components* on page 9-3011.
- *9.8 Differences between the VE hardware and the system models* on page 9-3017.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

9.1 VE - about

The Versatile Express (VE) FVPs are functionally accurate system models for software execution. A range of VE FVPs are supplied as standalone products and as examples in Fast Models.

For a list of the VE Platform FVPs and the instances in them, see *VE Platform FVPs* in the FVP Reference Guide.

LISA+ source and project files for the VE FVP examples are available in the `$PVLIB_HOME/examples/LISA/FVP_VE/` directory.

Arm produces the VE hardware development platform. The Motherboard Express *µAdvanced Technology Extended* (ATX) V2M-P1 is the basis for an integrated software and hardware development system. This system is also based on the Arm *Symmetric MultiProcessor* (SMP) system architecture.

The VE FVPs are system models implemented in software. Each model contains:

- A virtual implementation of the Arm VE motherboard.
- A single daughterboard containing one or more Arm processors.
- Associated interconnections.

The motherboard provides:

- Peripherals for multimedia or networking environments.
- Access to motherboard peripherals and functions through a static memory bus to simplify access from daughterboards.
- High-performance PCI-Express slots for expansion cards.
- Consistent memory maps with different processor daughterboards that simplify software development and porting.
- Automatic detection and configuration of attached CoreTile Express and LogicTile Express daughterboards.
- Automatic shutdown for over-temperature or power supply failure.
- No system power-on for unconfigurable daughterboards.
- Power sequencing of system.
- Drag and drop file updating of configuration files.
- Support of either a 12V power-supply unit or an external ATX power supply.
- Support of FPGA and processor daughterboards to provide custom peripherals, early access to processor designs, or production test chips.

Note

Arm bases the models on the VE platform memory map, but does not intend them to be accurate representations of a specific VE hardware revision. The VE FVPs support selected peripherals. The models are sufficiently complete and accurate to boot the same operating system images as the VE hardware.

VE FVPs provide functionally accurate models for software execution. However, the models sacrifice timing accuracy to increase simulation speed. Key deviations from hardware are:

- Approximate timing.
- Simplified buses.
- No implementations for processor caches and the related write buffers.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

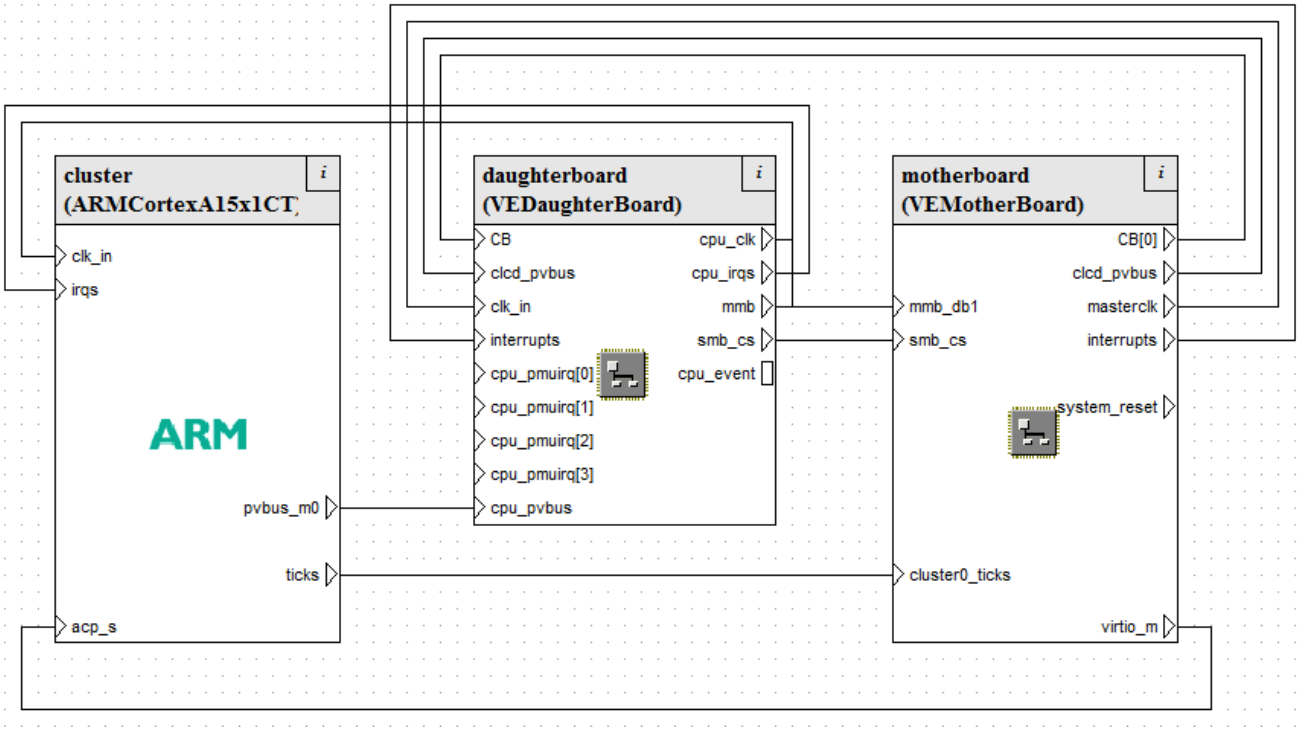

Figure 9-1 Top-level block diagram of a VE model

Related information

Getting Started with Fixed Virtual Platforms, Fixed Virtual Platforms FVP Reference Guide
Motherboard Express µATX V2M-P1 Technical Reference Manual

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

9.2 VE memory map for Cortex®-A series

The global memory map for the Cortex-A series VE model is based on the hardware Versatile Express RS1 memory map with the RS2 extensions.

Note

The VE FVP implementation of memory does not require the memory controller to have the correct values. If you run applications on hardware, ensure that the memory controller is set up properly. Otherwise, applications that run on the FVP might fail on hardware.

Table 9-1 Cortex-A series platform model memory map

Name	Modeled	Address range	Size
NOR FLASH0 (CS0)	Yes	0x00_00000000-0x00_03FFFFFF	64MB
Reserved	-	0x00_04000000-0x00_07FFFFFF	64MB
NOR FLASH0 alias (CS0)	Yes	0x00_08000000-0x00_0BFFFFFF	64MB
NOR FLASH1 (CS4)	Yes	0x00_0C000000-0x00_0FFFFFFF	64MB
Unused (CS5)	-	0x00_10000000-0x00_13FFFFFF	-
PSRAM (CS1) - unused	No	0x00_14000000-0x00_17FFFFFF	-
Peripherals (CS2). See *CS2 region peripheral memory map* on page 9-3003, below.	Yes	0x00_18000000-0x00_1BFFFFFF	64MB
Peripherals (CS3). See *CS3 region peripheral memory map* on page 9-3003, below.	Yes	0x00_1C000000-0x00_1FFFFFFF	64MB
CoreSight and peripherals	No	0x00_20000000-0x00_2CFFFFFF[ak]	-
Graphics space	No	0x00_2D000000-0x00_2D00FFFF	-
System SRAM	Yes	0x00_2E000000-0x00_2EFFFFFF	64KB
Ext AXI	No	0x00_2F000000-0x00_7FFFFFFF	-
4GB DRAM (in 32-bit address space)[al]	Yes	0x00_80000000-0x00_FFFFFFFF	2GB
Unused	-	0x01_00000000-0x07_FFFFFFFF	-
4GB DRAM (in 36-bit address space)[al]	Yes	0x08_00000000-0x08_FFFFFFFF	4GB
Unused	-	0x09_00000000-0x7F_FFFFFFFF	-
4GB DRAM (in 40-bit address space)[al]	Yes	0x80_00000000-0xFF_FFFFFFFF	4GB

The model has a `secure_memory` option. When you enable this option, the memory map changes for a number of peripherals.

[ak] The private peripheral region address `0x2c000000` is mapped in this region. You can use the parameter PERIPHBASE to map the peripherals to a different address.

[al] The model contains a single 4GB block of DRAM, which is aliased across the three different regions. In other words, it can be accessed at three different physical addresses, which are all mapped to the same area of DRAM. For example, a write to address `0x00_80000000` will be visible to reads at address `0x80_00000000`. The lowest of the physical address regions is only 2GB in size.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 9-2 CS2 region peripheral memory map for secure_memory option

Peripheral	Address range	Functionality with secure_memory enabled
NOR FLASH0 (CS0)	0x00_00000000-0x00_0001FFFF	Secure RO, aborts on non-secure accesses.
Reserved	0x00_04000000-0x00_0401FFFF	Secure SRAM, aborts on non-secure accesses.
NOR FLASH0 alias (CS0)	0x00_08000000-0x00_7DFFFFFF	Normal memory map, aborts on secure accesses.
Ext AXI	0x00_7e000000-0x00_7FFFFFFF	Secure DRAM, aborts on non-secure accesses.
4GB DRAM (in 32-bit address space)	0x00_80000000-0xFF_FFFFFFFF	Normal memory map, aborts on secure accesses.

Table 9-3 CS2 region peripheral memory map

Peripheral	Modeled	Address range	Size	GIC Int[am]
VRAM - aliased	Yes	0x00_18000000-0x00_19FFFFFF	32MB	-
Ethernet (SMSC 91C111)	Yes	0x00_1A000000-0x00_1AFFFFFF	16MB	47
USB - unused	No	0x00_1B000000-0x00_1BFFFFFF	16MB	-

Table 9-4 CS3 region peripheral memory map

Peripheral	Modeled	Address range	Size	GIC Int[am]
Local DAP ROM	No	0x00_1C000000-0x00_1C00FFFF	64KB	-
VE System Registers	Yes	0x00_1C010000-0x00_1C01FFFF	64KB	-
System Controller (SP810)	Yes	0x00_1C020000-0x00_1C02FFFF	64KB	-
TwoWire serial interface (PCIe)	No	0x00_1C030000-0x00_1C03FFFF	64KB	-
AACI (PL041)	Yes	0x00_1C040000-0x00_1C04FFFF	64KB	43
MCI (PL180)	Yes	0x00_1C050000-0x00_1C05FFFF	64KB	41, 42
KMI - keyboard (PL050)	Yes	0x00_1C060000-0x00_1C06FFFF	64KB	44
KMI - mouse (PL050)	Yes	0x00_1C070000-0x00_1C07FFFF	64KB	45
Reserved	-	0x00_1C080000-0x00_1C08FFFF	64KB	-
UART0 (PL011)	Yes	0x00_1C090000-0x00_1C09FFFF	64KB	37
UART1 (PL011)	Yes	0x00_1C0A0000-0x00_1C0AFFFF	64KB	38
UART2 (PL011)	Yes	0x00_1C0B0000-0x00_1C0BFFFF	64KB	39
UART3 (PL011)	Yes	0x00_1C0C0000-0x00_1C0CFFFF	64KB	40
Reserved	-	0x00_1C0D0000-0x00_1C0EFFFF	128KB	-
Watchdog (SP805)	Yes	0x00_1C0F0000-0x00_1C0FFFFF	64KB	32
Reserved	-	0x00_1C100000-0x00_1C10FFFF	64KB	-
Timer-0 (SP804)	Yes	0x00_1C110000-0x00_1C11FFFF	64KB	34
Timer-1 (SP804)	Yes	0x00_1C120000-0x00_1C12FFFF	64KB	35
Virtio block device	Yes	0x00_1C130000-0x00_1C13FFFF	64KB	74

[am] Use these interrupt signal values to program your interrupt controller. They are the SPI number plus 32. Add 32 to the interrupt numbers from the peripherals to form the interrupt number that the GIC sees. GIC interrupts 0-31 are for internal use.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 9-4 CS3 region peripheral memory map (continued)

Peripheral	Modeled	Address range	Size	GIC Int[am]
Virtio P9 device	Yes	0x00_1C140000-0x00_1C14FFFF	64KB	75
Reserved	-	0x00_1C130000-0x00_1C15FFFF	192KB	-
TwoWire serial interface (DVI) - unused	No	0x00_1C160000-0x00_1C16FFFF	64KB	-
Real-time Clock (PL031)	Yes	0x00_1C170000-0x00_1C17FFFF	64KB	36
Reserved	-	0x00_1C180000-0x00_1C19FFFF	128KB	-
CF Card - unused	No	0x00_1C1A0000-0x00_1C1AFFFF	64KB	
Reserved	-	0x00_1C1B0000-0x00_1C1EFFFF	256KB	-
Color LCD Controller (PL111)	Yes	0x00_1C1F0000-0x00_1C1FFFFF	64KB	46
Reserved	-	0x00_1C200000-0x00_1FFFFFFF	64KB	-

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

9.3 VE memory map for Cortex®-R series

The Versatile Express RS1 memory map with the RS2 extensions is the base of the global memory map for the Cortex-R series platform model.

Table 9-5 Cortex-R series VE FVP memory map

Memory	Modeled	Address range
DRAM	Yes	0x00000000-0x3FFFFFFF
FLASH0	Yes	0x40000000-0x43FFFFFF
FLASH1	Yes	0x44000000-0x47FFFFFF
PSRAM	Yes	0x48000000-0x4BFFFFFF
RAM	No	0x4C000000-0x4FFFFFFF
PL390 GIC CPU Interface [an]	Yes	0xAE000000-0xAE000FFF
PL390 GIC Distributor [an]	Yes	0xAE001000-0xAE001FFF
VE System Registers	Yes	0xB0000000–0xB000FFFF
SP810	Yes	0xB0010000–0xB001FFFF
PL041 AACI	Yes	0xB0040000–0xB004FFFF
PL180 MCI	Yes	0xB0050000–0xB005FFFF
PL050 KMI0	Yes	0xB0060000–0xB006FFFF
PL050 KMI1	Yes	0xB0070000–0xB007FFFF
PL011 UART0	Yes	0xB0090000–0xB009FFFF
PL011 UART1	Yes	0xB00A0000–0xB00AFFFF
PL011 UART2	Yes	0xB00B0000–0xB00BFFFF
PL011 UART3	Yes	0xB00C0000–0xB00CFFFF
SP805 WATCHDOG	Yes	0xB00F0000–0xB00FFFFF
TIMER_0_1	Yes	0xB0110000–0xB011FFFF
TIMER_2_3	Yes	0xB0120000–0xB012FFFF
PL031 Real Time Clock	Yes	0xB0170000–0xB017FFFF
Compact Flash	No	0xB01A0000–0xB01AFFFF
PL011 UART4	Yes	0xB01B0000–0xB01BFFFF
PL111 CLCD	Yes	0xB01F0000–0xB01FFFFF [ao]
RAM	No	0xB4000000–0xBBFFFFFF
Video RAM	Yes	0xBC000000–0xBDFFFFFF
Ethernet (SMSC 91C111)	Yes	0xBE000000–0xBEFFFFFF
USB	No	0xBF000000–0xBFFFFFFF

[an] Cortex-R4 and Cortex-R5 models only.
[ao] For Cortex-R4 and Cortex-R5 models, the range is 0xA0000000–0xA0010000.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

9.4 VE - interrupt assignments for Cortex®-A series

The platform routes the following *Shared Peripheral Interrupts* (SPIs) to the GIC.

Table 9-6 SPI GIC assignments

IRQ ID	SPI offset	Device
32	0	Watchdog, SP805
34	2	Dual timer 0/1, SP804
35	3	Dual timer 2/3, SP804
36	4	Real-time Clock, PL031
37	5	UART0, PL011
38	6	UART1, PL011
39	7	UART2, PL011
40	8	UART3, PL011
41	9	MCI, PL180, MCIINTR0
42	10	MCI, PL180, MCIINTR1
43	11	AACI, PL041
44	12	KMI - Keyboard, PL050
45	13	KMI - Mouse, PL050
46	14	Color LCD Controller, PL111
47	15	Ethernet, SMSC 91C111
74	42	Virtio block device
75	43	Virtio P9 device
92	60	CPU 0 PMU
93	61	CPU 1 PMU
94	62	CPU 2 PMU
95	63	CPU 3 PMU
117	85	HDLCD

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

9.5 VE - interrupt assignments for Cortex®-R series

This section describes the interrupt assignments.

Table 9-7 Interrupt assignments for Cortex-R4 and Cortex-R5

GIC IRQ ID	SPI offset	Device
32	0	Watchdog, SP805
34	2	Dual timer 0/1, SP804
35	3	Dual timer 2/3, SP804
36	4	Real-time Clock, PL031
41	9	MCI, PL180, MCIINTR0
42	10	MCI, PL180, MCIINTR1
43	11	AACI, PL041
44	12	KMI - Keyboard, PL050
45	13	KMI - Mouse, PL050
47	15	Ethernet, SMSC 91C111
75	16	Level 2 Cache Controller, PL310 Combined interrupt
76	14	Color LCD Controller, PL111
96	5	UART0, PL011
97	6	UART1, PL011
98	7	UART2, PL011
99	8	UART3, PL011

Table 9-8 Interrupt assignments for Cortex-R7 and Cortex-R8

GIC IRQ ID	SPI offset	Device
32	0	Watchdog, SP805
34	2	Dual timer 0/1, SP804
35	3	Dual timer 2/3, SP804
36	4	Real-time Clock, PL031
37	5	UART0, PL011
38	6	UART1, PL011
39	7	UART2, PL011
40	8	UART3, PL011
41	9	MCI, PL180, MCIINTR0
42	10	MCI, PL180, MCIINTR1
43	11	AACI, PL041
44	12	KMI - Keyboard, PL050
45	13	KMI - Mouse, PL050

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 9-8 Interrupt assignments for Cortex-R7 and Cortex-R8 (continued)

GIC IRQ ID	SPI offset	Device
46	14	Color LCD Controller, PL111
47	15	Ethernet, SMSC 91C111

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

9.6 VE parameters

This section describes the VE FVP instantiation parameters.

This section contains the following subsections:

- *9.6.1 VE instantiation parameters* on page 9-3009.
- *9.6.2 VE secure memory parameters* on page 9-3009.
- *9.6.3 VE switch S6* on page 9-3009.

9.6.1 VE instantiation parameters

This section describes the instantiation parameters for VE models.

Table 9-9 VE instantiation parameters

Component	Parameter	Type	Allowed values	Default value	Description
`ve_sysregs`	`user_switches_value`	Integer	See *9.6.3 VE switch S6* on page 9-3009.	`0`	Switch S6 setting.
`flashloader0`	`fname`	String	Valid filename	`""`	Path to flash image file.
`flashloader1`	`fname`	String	Valid filename	`""`	Path to flash image file.
`mmc`	`p_mmc_file`	String	Valid filename	`mmc.dat`	Multimedia card filename.
`pl111_clcd`	`pixel_double_limit`	Integer	-	`0x12c`	Sets threshold in horizontal pixels below which pixels sent to framebuffer are doubled in size in both dimensions.
`sp810_sysctrl`	`use_s8`	Boolean	`true`, `false`	`false`	Indicates whether to read `boot_switches_value`.

9.6.2 VE secure memory parameters

This section describes the VE FVP secure memory parameters that you can change when you start the model.

Table 9-10 VE secure memory parameters

Name	Type	Allowed values	Default value	Description
`daughterboard.secure_memory`	Boolean	`true`, `false`	`false`	**`false`** The platform behaves as before. **`true`** The address space is segregated according to the security mode of the core. Some memory blocks near the bottom of the address space are available to Secure transactions only. The rest of the address space is available to Non-secure transactions only.

9.6.3 VE switch S6

This section describes the behavior and default positions of the VE system model switch.

Switch S6 is equivalent to the Boot Monitor configuration switch on the VE hardware.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

If you have the standard Arm Boot Monitor flash image loaded, the setting of switch S6-1 changes what happens on model reset. Otherwise, the function of switch S6 is implementation dependent.

To write the switch position directly to the S6 parameter in the model, you must convert the switch settings to an integer value from the equivalent binary, where 1 is on and 0 is off.

Table 9-11 Default positions of VE system model switch

Switch	Default position	Function in default position
S6-1	OFF	Displays prompt permitting Boot Monitor command entry after system start.
S6-2	OFF	See STDIO redirection, below.
S6-3	OFF	See STDIO redirection, below.
S6-4 to S6-8	OFF	Reserved for application use.

If S6-1 is in the ON position, the Boot Monitor executes the boot script that was loaded into flash. If there is no script, the Boot Monitor prompt is displayed.

The settings of S6-2 and S6-3 affect STDIO source and destination on model reset.

Table 9-12 STDIO redirection

S6-2	S6-3	Output	Input	Description
OFF	OFF	UART0	UART0	STDIO autodetects whether to use semihosting I/O or a UART. If a debugger is connected, STDIO is redirected to the debugger output window, otherwise STDIO goes to UART0.
OFF	ON	UART0	UART0	STDIO is redirected to UART0, regardless of semihosting settings.
ON	OFF	CLCD	Keyboard	STDIO is redirected to the CLCD and keyboard, regardless of semihosting settings.
ON	ON	CLCD	UART0	STDIO output is redirected to the LCD and input is redirected to UART0, regardless of semihosting settings.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

9.7 VE - components

A complete model implementation of the VE platform includes both VE-specific components and generic components, such as buses and timers.

To see a list of all the component instances in the model, run it with the `--list-instances` option.

The generic components are documented in *Chapter 3 Fast Models components* on page 3-87.

This section contains the following subsections:

- *9.7.1 VEVisualisation component* on page 9-3011.
- *9.7.2 VE_SysRegs component* on page 9-3015.

9.7.1 VEVisualisation component

This section describes the VEVisualisation component.

VEVisualisation - about

This component can generate events from the host mouse or keyboard when the visualization window is in focus. For example, you can toggle the switch elements from the visualization window.

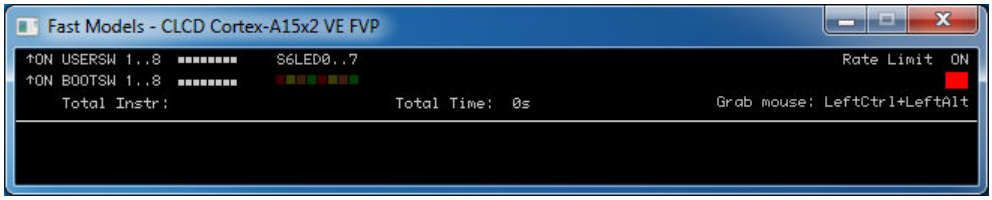

Figure 9-2 Startup VE FVP CLCD visualization window

When a suitable application or system image loads, and configures the PL111_CLCD controller registers, the window expands to show the contents of the frame buffer.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

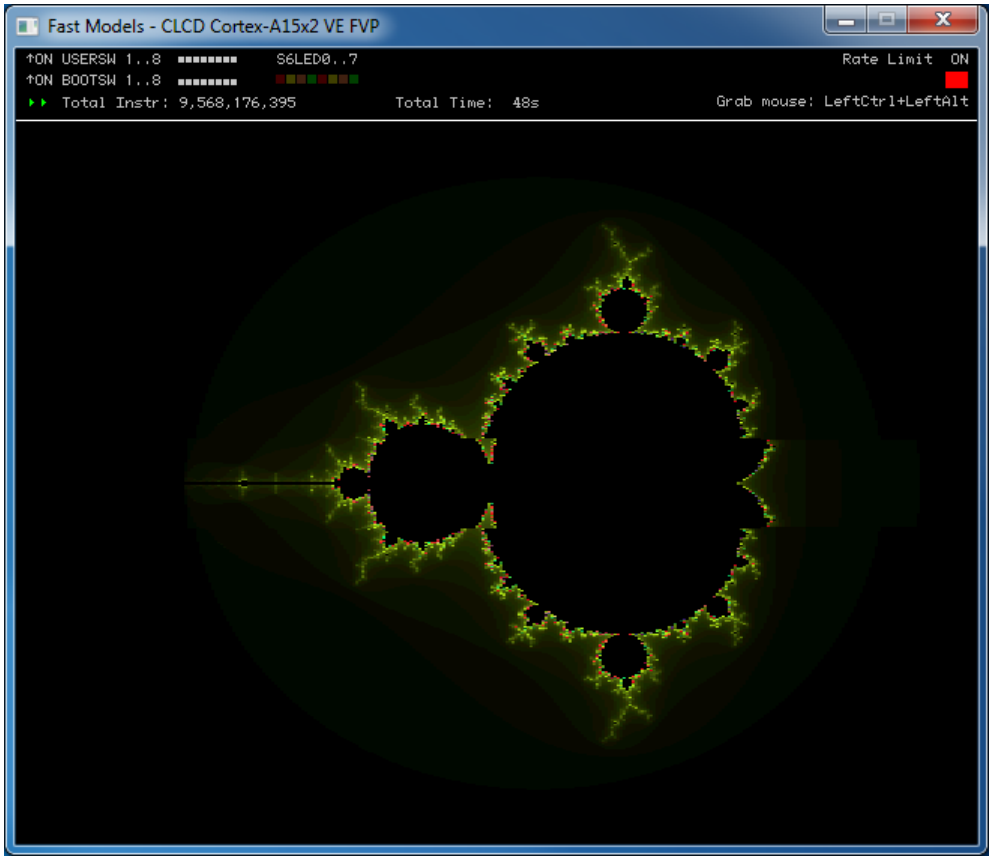

Figure 9-3 VE FVP CLCD with brot.axf image

The VEVisualisation LISA+ component can be found in the `$PVLIB_HOME/examples/LISA/FVP_VE/LISA/` directory.

Note

Using this component can reduce simulation performance. Use the `rate_limit-enable` parameter to control simulation speed.

Related references

VEVisualisation - parameters on page 9-3013

Related information

Fixed Virtual Platforms FVP Reference Guide

VEVisualisation - ports

This section describes the VEVisualisation component ports.

Table 9-13 VEVisualisation ports

Name	Protocol	Type	Description
`boot_switch`	ValueState	Slave	Provides state for the eight Boot DIP switches on the right side of the CLCD status bar.
`clock_50Hz`	ClockSignal	Slave	50Hz clock input.
`cluster0_ticks[4]`	InstructionCount	Slave	Connection from processor model in cluster 0 to show its current instruction count.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 9-13 VEVisualisation ports (continued)

Name	Protocol	Type	Description
`cluster1_ticks[4]`	InstructionCount	Slave	Connection from processor model in cluster 1 to show its current instruction count.
`daughter_leds`	ValueState	Slave	A read/write port to read and set the value of the LEDs. 1 bit per LED, LSB left-most, up to 32 LEDs available. The LEDs appear only when parameter `daughter_led_count` is set to nonzero.
`daughter_user_switches`	ValueState	Slave	A read port to return the value of the daughter user switches. Write to this port to set the value of the switches, and use during reset only. LSB is left-most, up to 32 switches available.
`keyboard`	KeyboardStatus	Master	Output port providing key change events when the visualization window is in focus.
`lcd`	LCD	Slave	Connection from a CLCD controller for visualization of the frame buffer.
`lcd_layout`	LCDLayoutInfo	Master	Layout information for alphanumeric LCD display.
`leds`	ValueState	Slave	Displays state using the eight colored LEDs on the status bar.
`mouse`	MouseStatus	Master	Output port providing mouse movement and button events when the visualization window is in focus.
`touch_screen`	MouseStatus	Master	Provides mouse events when the visualization window is in focus.
`user_switches`	ValueState	Slave	Provides state for the eight User DIP switches on the left side of the CLCD status bar, equivalent to switch S6 on VE hardware.

Related references

2.4.6 KeyboardStatus protocol on page 2-72
2.4.7 LCD protocol on page 2-72
2.4.10 MouseStatus protocol on page 2-74

VEVisualisation - parameters

This section describes the configuration parameters.

The syntax to use in a configuration file or on the command line is:

`motherboard.vis.`*`parameter=value`*

Note

Setting the `rate_limit-enable` parameter to `true` (the default) prevents the simulation from running too fast on fast workstations and enables timing loops and mouse actions to work correctly. However, it reduces the overall simulation speed. If your priority is high simulation speed, set `rate_limit-enable` to `false`.

Table 9-14 VEVisualisation parameters

Name	Type	Allowed values	Default value	Description
`cluster0_name`	`string`	-	Cluster0	Label for cluster 0 performance values.
`cluster1_name`	`string`	-	Cluster1	Label for cluster 1 performance values.
`cpu_name`	`string`	-	-	Processor name displayed in window title.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 9-14 VEVisualisation parameters (continued)

Name	Type	Allowed values	Default value	Description
daughter_led_count	int	0-32	0	Set to nonzero to display up to 32 LEDs. See the daughter_leds port.
daughter_user_switch_count	int	0-32	0	Set this parameter to display up to 32 switches. See the daughter_user_switches port.
disable_visualisation	bool	true, false	false	Disable the VEVisualisation component on model startup.
rate_limit-enable	bool	true, false	true	Restrict simulation speed so that simulation time more closely matches real time rather than running as fast as possible.
recorder.checkInstructionCount	bool	true, false	true	Check instruction count in recording file against actual instruction count during playback.
recorder.playbackFileName	string	-	""	Playback filename (empty string disables playback).
recorder.recordingFileName	string	-	""	Recording filename (empty string disables recording).
recorder.recordingTimeBase	int	-	0x5F5E100	Timebase in 1/s (relative to the master clock (where 100000000 means 10 nanoseconds resolution simulated time for a 1Hz master clock)) for recording (higher values give higher time resolution, playback timebase is always taken from the playback file).
recorder.verbose	int	-	0x0	Enable verbose messages (1=normal, 2=even more).
trap_key	int	Valid ATKeyCode key value[ap]	74, 0x4A[aq]	Trap key that works with left **Ctrl** key to toggle mouse display.
window_title	string	-	"Fast Models - CLCD %cpu%"	Window title (cpu_name replaces %cpu%).

VEVisualisation - verification and testing

This component passes tests by use as an I/O device for booting Linux and other operating systems.

VEVisualisation - performance

Arm expects the elements in the status bar to have little effect on the performance of PV systems. However, applications that often redraw the contents of the frame buffer might incur overhead through GUI interactions on the host OS.

VEVisualisation - library dependencies

This component relies on the *Simple DirectMedia Layer* (SDL) libraries, specifically libSDL2-2.0.so.0.4.0.

[ap] See the header file, $PVLIB_HOME/include/components/KeyCode.h for a list of ATKeyCode values.
[aq] This is equivalent to the left **Alt** key, so pressing Left Alt and Left Ctrl simultaneously toggles the mouse display.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

This library is bundled with the Model Library and is also available as an rpm for Red Hat Enterprise Linux. On Windows, the library is called `SDL2.dll`.

Related information

Simple DirectMedia Layer Cross-platform Development Library

9.7.2 VE_SysRegs component

This section describes the VE system registers component.

VE_SysRegs - about

This LISA+ component is a model of the VE status and system control registers.

VE_SysRegs - ports

This section describes the ports.

Table 9-15 VE_SysRegs ports

Name	Protocol	Type	Description
`cb[0-1]`	`VECBProtocol`	Master	The *Configuration Bus* (CB) controls the power and reset sequence.
`clock_24Mhz`	`ClockSignal`	Slave	Reference clock for internal counter register.
`clock_100Hz`	`ClockSignal`	Slave	Reference clock for internal counter register.
`clock_CLCD`	`ClockRateControl`	Master	The clock for the LCD controller.
`lcd`	`LCD`	Master	Multimedia bus interface output to the LCD.
`leds`	`ValueState`	Master	Displays state of the SYS_LED register using the eight colored LEDs on the status bar.
`mmb[0-2]`	`LCD`	Slave	Multimedia bus interface input.
`mmc_card_present`	`StateSignal`	Slave	Indicates the presence of a *MultiMedia Card* (MMC) image.
`pvbus`	`PVBus`	Slave	Slave port for connection to PV bus master/decoder.
`system_reset`	`Signal`	Master	Signal to the platform a complete system reset. Writes to the System Configuration registers can trigger the reset signal.
`user_switches`	`ValueState`	Master	Provides state for the eight User DIP switches on the left side of the CLCD status bar, equivalent to switch S6 on VE hardware.

VE_SysRegs - parameters

This section describes the parameters.

Table 9-16 VE_SysRegs parameters

Name	Type	Default value	Description
`exit_on_shutdown`	`bool`	`false`	Used to shut down the system. When `true`, if software uses the `SYS_CFGCTRL` function `SYS_CFG_SHUTDOWN`, then the simulator shuts down and exits.[ar]
`mmbSiteDefault`	`int`	`0x1`	Default MultiMedia Bus (MMB) source (0=motherboard, 1=daughterboard 1, 2=daughterboard 2).
`sys_proc_id0`	`int`	`0x0c000000`	Processor ID register at CoreTile Express Site 1.
`sys_proc_id1`	`int`	`0xff000000`	Processor ID at CoreTile Express Site 2.

[ar] For more information on the `SYS_CFGCTRL` function values, see the *Motherboard Express µATX V2M-P1 Technical Reference Manual*.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

Table 9-16 VE_SysRegs parameters (continued)

Name	Type	Default value	Description
tilePresent	bool	true	Tile fitted.
user_switches_value	int	0	User switches.

VE_SysRegs - registers

This section describes the configuration registers.

Table 9-17 VE_SysRegs registers

Name	Offset	Access	Description
SYS_ID	0x00	Read/write	System identity.
SYS_SW	0x04	Read/write	Bits[7:0] map to switch S6.
SYS_LED	0x08	Read/write	Bits[7:0] map to user LEDs.
SYS_100HZ	0x24	Read only	100Hz counter.
SYS_FLAGS	0x30	Read/write	General purpose flags.
SYS_FLAGSCLR	0x34	Write only	Clear bits in general purpose flags.
SYS_NVFLAGS	0x38	Read/write	General purpose non-volatile flags.
SYS_NVFLAGSCLR	0x3C	Write only	Clear bits in general purpose non-volatile flags.
SYS_MCI	0x48	Read only	MCI.
SYS_FLASH	0x4C	Read/write	Flash control.
SYS_CFGSW	0x58	Read/write	Boot select switch.
SYS_24MHZ	0x5C	Read only	24MHz counter.
SYS_MISC	0x60	Read/write	Miscellaneous control flags.
SYS_DMA	0x64	Read/write	DMA peripheral map.
SYS_PROCID0	0x84	Read/write	Processor ID.
SYS_PROCID1	0x88	Read/write	Processor ID.
SYS_CFGDATA	0xA0	Read/write	Data to be read/written from/to motherboard controller.
SYS_CFGCTRL	0xA4	Read/write	Control data transfer to motherboard controller.
SYS_CFGSTAT	0xA8	Read/write	Status of data transfer to motherboard.

VE_SysRegs - verification and testing

This component was tested as part of the Versatile Express model.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

9.8 Differences between the VE hardware and the system models

This section describes features of the hardware that the models do not implement, or implement with significant differences.

This section contains the following subsections:

- *9.8.1 Memory map* on page 9-3017.
- *9.8.2 Memory aliasing* on page 9-3017.
- *9.8.3 VE hardware features absent* on page 9-3017.
- *9.8.4 VE hardware features different* on page 9-3017.
- *9.8.5 Timing considerations for the VE FVPs* on page 9-3018.

9.8.1 Memory map

The model represents the memory map of the hardware VE platform, but is not an accurate representation of a specific revision.

The memory map in the supplied model is sufficiently complete and accurate to boot the same operating system images as for the VE hardware.

In the memory map, memory regions that are not explicitly occupied by a peripheral or by memory are unmapped. This includes regions otherwise occupied by a peripheral that is not implemented, and those areas that are documented as reserved. Accessing these regions from the host processor results in the model presenting a warning.

9.8.2 Memory aliasing

The model implements address-space aliasing of the DRAM. This means that the same physical memory locations are visible at different addresses.

The lower 2GB of the DRAM is accessible at `0x00_80000000`. The full 4GB of DRAM is accessible at `0x08_00000000` and again at `0x80_00000000`. The aliasing of DRAM then repeats from `0x81_00000000` up to `0xFF_FFFFFFFF`.

9.8.3 VE hardware features absent

These FVPs do not implement the following features of the hardware:

- Two-wire serial bus interfaces.
- USB interfaces.
- PCI Express interfaces.
- Compact flash.
- *Digital Visual Interface* (DVI).
- Debug and test interfaces.
- *Dynamic Memory Controller* (DMC).
- *Static Memory Controller* (SMC).

Related references

9.2 VE memory map for Cortex®-A series on page 9-3002

9.8.4 VE hardware features different

These Fixed Virtual Platforms only partially implement some features of the hardware.

The partially implemented features might not work as you expect. Check the model release notes for the latest information.

Sound

The VE FVPs implement the PL041 AACI PrimeCell and the audio CODEC as in the VE hardware, but with a limited number of sample rates.

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential

9.8.5 Timing considerations for the VE FVPs

The Rate Limit feature matches simulation time to wall-clock time.

The *Fixed Virtual Platforms* (FVPs) provide an environment that enables running software applications in a functionally-accurate simulation. However, because of the relative balance of fast simulation speed over timing accuracy, there are situations where the models might behave unexpectedly.

When code interacts with real world devices like timers and keyboards, data arrives in the modeled device in real world, or wall clock, time, but simulation time can run much faster than the wall clock. This means that a single key press might register as several repeated key presses, or a single mouse click incorrectly becomes a double click.

Enabling Rate Limit, either using the Rate Limit button in the CLCD display, or the `rate_limit-enable` model instantiation parameter, forces the model to run at wall-clock time. This avoids issues with two clocks running at significantly different rates. For interactive applications, Arm recommends enabling Rate Limit.

Related references

1.3 Fast Models accuracy on page 1-15

1.4 Processor implementation on page 1-19

Copyright © 2014–2020 Arm Limited or its affiliates. All rights reserved.
Non-Confidential